令和７年度「地方やデジタル分野における専修学校理系転換等推進事業」

教員研修資料

本教員研修資料は、文部科学省の教育政策推進事業委託費による委託事業として、一般社団法人全国専門学校情報教育協会が実施した令和７年度「地方やデジタル分野における専修学校理系転換等推進事業」の成果物です。

情報成長分野の教育プログラム整備と教員育成による学科転換・新設推進事業

目次

ビッグデータ技術応用 6

実習 1　Python による顧客データ分析とレポーティング 6

教員用導入ガイド 6

教員用指導ガイド 9

実習 2　アクセスログ ETL 処理と BI ツールによる可視化 16

教員用導入ガイド 16

教員用指導ガイド 19

実習 3　Spark による大規模データ処理 25

教員用導入ガイド 25

教員用指導ガイド 29

人工知能（AI）技術応用 36

実習 1　機械学習で手書き数字を認識 36

導入編　教員用指導ガイド 36

Part 1 ：環境準備とデータ確認　教員用指導ガイド 38

Part 2 ：3 つのアルゴリズム実装　教員用指導ガイド 43

Part 3 ：結果の比較と評価　教員用指導ガイド 47

Part 4 ：実データでの検証　教員用指導ガイド 49

実習 2　画像認識 AI で犬と猫を分類 51

導入編　教員用指導ガイド 51

Part 1 ：データ準備と CNN 入門　教員用指導ガイド 54

Part 2 ：CNN モデルの構築と学習　教員用指導ガイド 56

Part 3 ：モデルの評価と改善　教員用指導ガイド 59

Part 4 ：転移学習と実データテスト　教員用指導ガイド 62

実習 3：自然言語処理で感情分析 AI を作る 64

導入編　教員用指導ガイド 64

Part 1 ：環境準備とデータ確認　教員用指導ガイド 67

Part 2 ：RNN/LSTM モデルの構築　教員用指導ガイド 70

Part 3 ：モデルの評価と可視化　教員用指導ガイド 72

Part 4 ：事前学習モデル（BERT）の活用　教員用指導ガイド 74

IoT 技術応用 78

実習 1　デバイス制御 78

導入編　教員用指導ガイド 78

Part1: LED 制御の基礎　教員用指導ガイド 92

Part2: センサーでデータ取得 教員用指導ガイド 104
Part3: 条件による制御 教員用指導ガイド 121
Part4: データ記録と表示 教員用指導ガイド 143

実習 2 クラウド接続 172
導入編 教員用指導ガイド 172
Part1: Wi-Fi 接続とネットワーク基礎 教員用指導ガイド 176
Part 2 :センサーデータ取得とシリアル通信 教員用指導ガイド 182
Part 3 ：クラウドサービス連携 教員用指導ガイド 188
Part 4 ：データ可視化と分析 教員用指導ガイド 194

実習 3 システム統合 200
導入編 教員用指導ガイド 200
Part 1 ：Python とデータ分析の基礎 教員用指導ガイド 203
Part 2 ～ 4 教員用指導ガイド 207

ビッグデータ技術応用

ビッグデータ技術応用
実習１　Python による顧客データ分析とレポーティング
教員用導入ガイド

§ 🎯 実習の概要と指導目標

本実習（Exercise 1）は、応用教材の最初のステップとして、Python を用いたデータ分析の基本的な流れ（データ取得→前処理→分析→可視化→レポーティング）を学生に体験させることを目的とします。

基礎教材との関連：

- 第 1 章：ビッグデータ活用事例（顧客分析の位置づけ）
- 第 2 章：データ分析基礎（統計量、可視化、データクレンジング）
- 第 3 章：スクリプト言語による分析（pandas, matplotlib の基本操作）

🌟 **指導目標レベル：**

学生が、提供されたデータセットに対し、基本的なデータ分析プロセスを自力で（ワークブックを参照しながら）完遂し、簡単なレポートを作成できるレベルに到達させる。

💡 **全体を通した指導ポイント**

- **「なぜ？」を重視：**各処理（欠損値削除、グラフ選択など）の背景にある「目的」や「理由」を常に問いかけ、考えさせる
- **エラーは学びの機会：**コードのエラーを恐れず、メッセージを読み解き、解決するプロセスをサポートする。
- **ビジネス視点：**「この分析結果がビジネスにどう役立つか？」という視点を常に持たせる
- **ツールの使い方 ＜ 分析思考：**ツールの操作習得だけでなく、データから何を読み取り、どう解釈するかの「思考プロセス」を重視する

§ 💻 必要な環境とツール

📦 分析環境の選択肢とトラブルシューティング

🅰️ **オプション 1：Google Colaboratory（推奨）**

指導のポイント：

- 授業進行の安定性から、Colab 利用を強く推奨
- 初回授業で、Google アカウントでのログイン、ノートブック新規作成、ファイルアップロードの基本操作を丁寧にデモする
- 無料枠のリソース制限（特に RAM）について触れ、大規模データ処理には限界があることも示唆（実習 3 への伏線）

❌ **Colab よくあるトラブル**

症状

ログインできない、ノートブックが開けない、ファイルアップロードが失敗する

原因

Google アカウントの問題、ネットワーク不安定、ブラウザのキャッシュ/拡張機能

対処法

アカウント確認、ネットワーク再接続、ブラウザ再起動/キャッシュクリア/シークレットモード試行

🅱️ **オプション 2：ローカル環境（Jupyter Notebook）**

指導のポイント：

- 学生の環境差異によるトラブル発生を想定し、TA による個別サポート体制を準備
- Anaconda のインストール、仮想環境の作成（推奨）、主要ライブラリ（pandas, matplotlib, seaborn, openpyxl, pyarrow）のインストール手順を事前に配布

❌ **ローカル環境 よくあるトラブル**

症状

ModuleNotFoundError, Python バージョン不整合, Jupyter 起動失敗

原因

ライブラリ未インストール, Path 設定ミス, 複数 Python 環境の混在

対処法

pip install 実行確認, 環境変数確認, Anaconda Navigator からの起動試行, 仮想環境の再構築

§ 📂 使用するデータセット

📊 **Kaggle "E-Commerce Data"**

実習 1 では、Kaggle で公開されている data.csv を使用します。

URL： https://www.kaggle.com/datasets/carrie1/ecommerce-data

💡 **データ準備に関する指導ポイント**

- **Kaggle アカウント作成：** データダウンロードには Kaggle アカウントが必要です。事前に作成しておくようアナウンスしてください。
- **データ事前配布：** ネットワーク環境等でダウンロードできない学生のために、LMS 等で data.csv を事前配布しておくことを強く推奨します。
- **文字コード注意喚起：** Part 1 のワークブックで pd.read_csv() を使う際、encoding='ISO-8859-1' が必要になる可能性が高いです。なぜ必要なのか（UTF-8 以外の文字が含まれるため）を簡単に説明してください。
- **データの内容確認：** Part 1 の冒頭で、各列（特に CustomerID の欠損、Quantity のマイナス値）がどのような意味を持つのか、学生と一緒に確認・推測する時間を設けると良いでしょう。

❌ **データ読み込み よくあるトラブル**

症状

FileNotFoundError, UnicodeDecodeError（文字化け）

原因

ファイルパス間違い（Colab/Jupyter での配置場所）, エンコーディング指定ミス

対処法

Colab ならファイルがアップロードされているか確認, Jupyter なら.ipynb と同じフォルダにあるか確認。encoding='ISO-8859-1'（または'latin-1'）を試すよう指示

§ 📖 Part 別の学習内容と指導ポイント

各 Part の指導ポイントは、別途提供される「指導マニュアル（統合版）」[cite: 369-374]に詳述されていますが、ここでも概要を記載します。

- **Part 1：環境構築とデータ理解**
  - Colab 操作に慣れさせる。head, info, describe でデータの全体像を掴むことの重要性を強調。欠損値・異常値の「発見」をゴールとする。
- **Part 2：データクレンジングと前処理**
  - 「なぜ」その処理をするのか（分析目的との関連）を考えさせる。dropna, astype, 条件抽出の基本操作を定着させる。特徴量作成の意義を説明。
- **Part 3：探索的データ分析と可視化**
  - groupby の使い方を徹底的に練習。グラフ作成の目的（＝インサイト発見）を意識させる。グラフの種類選択の基本（時系列→折れ線、比較→棒など）を教える。
- **Part 4：統合分析（RFM）とレポート作成**
  - RFM 分析の概念とビジネス価値を説明。スコアリング（qcut）の考え方を解説。分析結果を「伝える」ことの重要性を強調し、レポート構成の基本を指導。

Part	内容	指導上の注意
Part 1	環境構築とデータ理解	Colab 操作とデータ読み込みに時間を割く
Part 2	データクレンジング	欠損値・異常値処理の判断理由を議論させる
Part 3	EDA と可視化	groupby とグラフ作成の反復練習
Part 4	RFM 分析とレポート	RFM の概念理解とレポート作成に時間をかける

ビッグデータ技術応用
実習 1　Python による顧客データ分析とレポーティング
教員用指導ガイド

§ 📋 このマニュアルについて

このマニュアルは、実習 1 を指導する教員・TA の方々向けに作成されています。

含まれる情報：

- よくあるトラブルと対処法（Part 別）
- 指導上の工夫とポイント
- 進捗確認チェックリスト
- 時間配分のアドバイス
- 評価の観点

§ 🎯 全体的な指導方針

基本姿勢

💡 **指導の基本原則**

- **励ます**：初心者を想定し、小さな成功を一緒に喜ぶ
- **待つ**：学生のペースに合わせ、焦らせない
- **実践重視**：理論より先に手を動かす。理解は後からついてくる
- **エラーを歓迎**：エラーは学びの機会。一緒に原因を探る
- **個別対応**：進度の差を認識し、遅れている学生をサポート

対象学生のレベル想定

- **前提知識**：プログラミング初心者〜中級者
- **Python 経験**：なし〜基本構文を少し知っている程度
- **統計知識**：平均・中央値程度は知っている
- **データ分析経験**：ほぼなし

§ 🔧 Part 1：環境構築とデータ理解

よくあるトラブルと対処法

❌ **トラブル 1-1：Google Colaboratory にアクセスできない**

症状

Google Colaboratory のページが開けない、またはログインできない

原因

- Google アカウントを持っていない
- 学校のネットワークで Google サービスがブロックされている
- ブラウザが古い

対処法

1. Google アカウントの作成を支援する（事前に案内しておくのが望ましい）
2. 学校の IT 部門に確認し、必要に応じてアクセス許可を得る
3. Chrome、Firefox、Edge の最新版を使用するよう指示
4. どうしてもアクセスできない場合は、ローカル環境（Jupyter）を使用

予防策

- 初回授業の 1 週間前にアクセステストを実施
- 事前に Google アカウント作成を宿題にする

❌ トラブル 1-2：ライブラリのインポートでエラー

症状

ModuleNotFoundError: No module named 'pandas' などのエラー

原因

- ローカル環境でライブラリがインストールされていない
- Google Colab ではこのエラーは通常発生しない

対処法

1. 新しいセルに !pip install pandas numpy matplotlib seaborn を実行
2. インストール後、カーネルを再起動
3. 再度 import を実行

❌ トラブル 1-3：データの読み込みが非常に遅い

症状

pd.read_excel() が 5 分以上経っても終わらない

原因

- インターネット接続が遅い
- Excel ファイル（約 20MB）のダウンロードに時間がかかっている

対処法

1. 辛抱強く待つ（通常 1-2 分、遅くても 5 分程度で完了）
2. 事前にダウンロードした CSV 版を使用する（より高速）
3. 教員が事前に変換した Parquet 形式ファイルを配布する

予防策

- 事前にデータをダウンロードし、Google Drive に配置しておく
- CSV 形式に変換したものを用意する

❌ トラブル 1-4：openpyxl がないというエラー

症状

ImportError: Missing optional dependency 'openpyxl'

原因

Excel ファイルを読むための openpyxl ライブラリがインストールされていない

対処法

1. 新しいセルに !pip install openpyxl を実行
2. インストール完了後、再度 pd.read_excel() を実行

指導のポイント

💡 Part 1 指導上の工夫

- **環境構築は丁寧に**：ここで躓くと以降の実習ができなくなる。十分な時間を取る
- **コードは一緒に入力**：最初は画面共有で一文字ずつ入力を見せる
- **成功体験を重視**：「Hello World」的な小さな成功を演出（データが表示された！）
- **専門用語は後回し**：DataFrame、Series 等の概念は詳しく説明せず、後で理解できれば OK

✓ Part 1 進捗確認チェックリスト

以下の項目を確認してから次に進みます：

- Python 環境（Colab or Jupyter）が起動している
- ライブラリが正常にインポートできた
- データを読み込めた（約 54 万行）
- df.head() でデータが表示できた
- データの形状（shape）を確認できた
- データ型（dtypes）を表示できた
- 欠損値の存在を確認できた

§ 🖌️ Part 2：データクレンジングと前処理

よくあるトラブルと対処法

❌ トラブル 2-1：dropna() 後もデータが変わらない

症状

欠損値を削除したはずなのに、行数が変わっていない

原因

pandas の操作は**デフォルトでは元の DataFrame を変更しない**（immutable）

対処法

以下のいずれかの方法を使う：

1. df = df.dropna() のように、結果を代入する
2. df.dropna(inplace=True) を使う（非推奨だが動作する）

予防策

ワークブックでは df_clean = df.dropna() のように新しい変数に代入する形で統一

❌ トラブル 2-2：データ型変換時のエラー

症状

ValueError: cannot convert float NaN to integer

原因

欠損値が残っている状態で整数型に変換しようとしている

対処法

1. 先に欠損値を削除してから型変換する
2. または Int64（大文字 I）を使う（欠損値を保持できる整数型）

❌ トラブル 2-3：削除しすぎてデータが極端に少なくなった

症状

欠損値や異常値を削除したら、元の 10%以下になってしまった

原因

- 複数の条件で削除を重ねすぎた
- 間違った列を指定して削除した

対処法

1. Part 1 からやり直す（データを再読み込み）
2. 各削除操作後に行数を確認する習慣をつける
3. 削除前に必ず len(df) で行数を記録

指導のポイント

💡 **Part 2 指導上の工夫**

- **判断の理由を説明**：「なぜこのデータを削除するのか」を必ず説明
- **削除の影響を可視化**：削除前後の行数を必ず表示させる
- **元データを保持**：df と df_clean を分けることの重要性を強調
- **ビジネス視点**：技術的な処理だけでなく、ビジネス的な意味を常に説明

✓ **Part 2 進捗確認チェックリスト**

- CustomerID 欠損行を削除できた
- Quantity ≤ 0 の行を削除できた
- UnitPrice ≤ 0 の行を削除できた
- CustomerID を整数型に変換できた
- TotalPrice 列を作成できた
- 日時から年・月・曜日・時間を抽出できた
- 最終的な欠損値が 0 になった
- 処理後のデータが 30 万行以上残っている

§ 📊 Part 3：探索的データ分析と可視化

よくあるトラブルと対処法

❌ **トラブル 3-1：グラフが表示されない**

症状

グラフ作成のコードを実行しても何も表示されない

原因

- plt.show() を忘れている
- Google Colab では通常不要だが、ローカル環境では必要

対処法

1. コードの最後に plt.show() を追加
2. Google Colab の場合、セルを再実行
3. Jupyter Notebook の場合、%matplotlib inline を先頭で実行

❌ **トラブル 3-2：groupby() の結果が理解できない**

症状

groupby()を実行しても期待した結果にならない

原因

- groupby()の構文を理解していない
- 集計関数（sum, mean, count 等）を指定していない

対処法

1. 小さなサンプルデータで練習させる
2. 「グループ化 → 集計」の 2 ステップで理解させる
3. 結果を毎回 print() で確認する習慣をつける

❌ トラブル 3-3：日本語がグラフで文字化けする

症状

グラフのタイトルや軸ラベルが□□□で表示される

原因

日本語フォントが設定されていない

対処法

ワークブックのコードでは英語表記を使用しているため、この問題は回避されています。もし日本語を使いたい場合：

1. Google Colab の場合、日本語フォントをインストール
2. または、タイトルや軸ラベルを英語にする

指導のポイント

💡 Part 3 指導上の工夫

- **グラフの意味を説明**：グラフを作ることが目的ではなく、何がわかるかが重要
- **発見を促す**：「このグラフから何が読み取れますか？」と問いかける
- **ビジネス視点**：「これは業務にどう活かせるか？」を常に考えさせる
- **比較させる**：複数のグラフを並べて比較する経験をさせる

✓ Part 3 進捗確認チェックリスト

- 顧客ごとの購買回数を集計できた
- 顧客ごとの総購入金額を集計できた
- ヒストグラムを作成できた
- 優良顧客を定義・抽出できた
- 人気商品トップ 10 を特定できた
- 国別売上を分析できた
- 月別売上トレンドをグラフ化できた
- 相関係数を計算できた
- 散布図を作成できた

§ 💎 Part 4：統合分析とレポート作成

よくあるトラブルと対処法

❌ トラブル 4-1：qcut() でエラーが出る

症状

```
ValueError: Bin edges must be unique
```

原因

データに同じ値が多く、5 分位に分割できない

対処法

1. qcut() に duplicates='drop' オプションを追加
2. または分位数を減らす（5→4 や 3）

❌ トラブル 4-2：RFM スコアの計算結果が理解できない

症状

RFM スコアが「555」なのに優良顧客に分類されない

原因

- スコアの計算ロジックを誤解している
- セグメント定義の閾値を間違えている

対処法

1. サンプルデータで具体例を示す
2. 各顧客の RFM スコアを 1 件ずつ確認させる
3. スコアリングのロジックを図解で説明

指導のポイント

💡 Part 4 指導上の工夫

- **統合の重要性**：Part 1-3 の分析が 1 つのストーリーになることを示す
- **RFM 分析の価値**：実務でも広く使われる手法であることを強調
- **レポートの質**：分析は正しくても、伝え方が悪いと価値がないことを説明
- **ビジネス提案**：データ分析者の仕事はレポートを書くことではなく、ビジネスを改善すること

✓ Part 4 進捗確認チェックリスト

- RFM 指標（R, F, M）を計算できた
- RFM スコア（1-5）を算出できた
- 顧客をセグメントに分類できた
- セグメント別の特徴を分析できた
- 主要な発見事項をまとめた
- 優良顧客の特徴を明確化できた
- セグメント別の施策を提案できた
- エグゼクティブサマリーを作成できた

§ 進度調整のコツ

💡 遅れている学生への対応

- **部分完成を認める**：全てのタスクを完了しなくても、重要部分ができていれば OK
- **ペア学習**：進んでいる学生と組ませる
- **オフィスアワー**：個別サポートの時間を設ける
- **優先順位**：「これだけは必須」というタスクを明示

§ ❓ 教員からのよくある質問

Q1: 学生のレベル差が大きい場合、どう対応すれば良いですか？

A1: 以下の方法を組み合わせてください

- **発展課題の活用：**進んでいる学生には難易度の高い課題を提示
- **ペア学習：**異なるレベルの学生を組ませる
- **最低限の到達点を明示：**「ここまでできれば OK」を明確に
- **オプション課題：**興味のある部分を深掘りできる選択課題を用意

Q2: プログラミング未経験者が多い場合、実習は可能ですか？

A2: 可能ですが、以下の対策が必要です

- **事前学習：**Python 基礎の簡単なチュートリアルを宿題に
- **時間を増やす：**各 Part に 1.5-2 倍の時間を確保
- **コピペを許容：**最初はコードをコピペして動かすことから始める
- **TA の増員：**サポート体制を手厚くする

Q3: オンライン授業での実施は可能ですか？

A3: 可能です。むしろ Google Colab を使うため相性が良いです

- **画面共有：**教員の画面を共有してコードを見せる
- **ブレイクアウトルーム：**小グループで助け合える環境を作る
- **Slack などのチャット：**質問しやすい環境を整備
- **録画：**授業を録画して復習できるようにする

ビッグデータ技術応用
実習２　アクセスログ ETL 処理と BI ツールによる可視化
教員用導入ガイド

§ 🎯 実習の概要と指導目標

本実習は、Web アクセスログを題材とし、ETL（Extract, Transform, Load）処理と BI ツールを用いたダッシュボード作成スキルを習得することを目的とします。
基礎教材の**第 3 章（Python 分析）**、**第 4 章（ツールとモニタリング）**、**第 6 章（データマート）**の内容を実践的に応用します。

指導目標：

- 非構造化データ（ログ）を Python で処理する基本的な流れを理解させる。
- ETL の各ステップ（抽出、変換、ロード）の目的と手法を理解させる。
- 分析目的に応じたデータマート設計の重要性を理解させる。
- 主要 BI ツール（Tableau Public, Power BI Desktop）の基本操作を習得させる。
- データ可視化を通じてインサイトを得る体験をさせる。

§ 💻 必要な環境とツール

学生には以下のツールの準備を指示してください。

ツール名	用途	備考
Python（+ pandas）	ETL 処理	実習 1 環境を継続（Google Colab 推奨）
Tableau Public	BI ツール①	無料，要アカウント登録，Mac/Win 対応
Power BI Desktop	BI ツール②	無料，Windows のみ

💡 指導上の TIPS：ツール選択

Tableau と Power BI の両方を扱うことで、ツールの違いを体験できますが、授業時間や学生の OS 環境に応じて、どちらか一方に絞ることも検討してください。

- **Mac ユーザーが多い場合：** Tableau Public が主体となります
- **Windows ユーザーが多い場合：** 両方、または Power BI Desktop が選択肢に入ります

§ 🔧 BI ツール導入トラブルシューティング

❌ トラブル：Tableau Public / Power BI Desktop がインストールできない

症状

インストーラーが起動しない、エラーが出る

原因

- PC のスペック不足（メモリ、ディスク容量）
- 管理者権限がない
- ネットワークの問題（ダウンロード失敗）
- OS バージョン非対応

対処法

1. 各ツールのシステム要件を確認させる
2. 管理者権限でインストーラーを実行させる
3. 再度ダウンロードを試みる
4. OS のアップデートを検討させる（難しい場合が多い）
5. 【最終手段】ペアワークでインストール済みの学生と組ませる

❌ トラブル：Tableau Public のアカウント登録ができない

症状

メールアドレス確認が届かない、登録エラ

原因

- 迷惑メールフォルダに入っている
- メールアドレスの入力ミス
- 学校のメールドメイン制限

対処法

1. 迷惑メールフォルダを確認させる
2. 別の個人メールアドレス（Gmail 等）で試させる

予防

事前にアカウント登録を済ませておくよう指示する

§ 📂 使用するデータセット

Web サーバーアクセスログを使用します。形式は Apache Combined Log Format を想定します。

データ準備方法

1. **サンプルデータ（推奨）：** 教材に含める sample_access.log を使用させるのが最もスムーズです。
2. **データ生成スクリプト（オプション）：** 提供する Python スクリプト generate_logs.py を使えば、学生が自分でデータ量や期間を指定してログファイルを生成できます。より実践的な演習を行いたい場合に活用できます。
3. **公開データセット（参考）：** NASA httpd logs などがありますが、形式の違いに対応する必要があるため、応用課題として提示するのが良いでしょう。

§ 💡 指導上の TIPS：データ生成スクリプト

データ生成スクリプトを使用する場合、その使い方（パラメータの変更方法など）を簡単にデモンストレーションすると良いでしょう。大量のデータを生成させ、ファイルサイズや処理時間の変化を体験させることも有効です。

§ 📖 Part 別の学習内容と指導ポイント

実習は 4 つの Part で構成されます。

- **Part 1：ログの理解とパース**
  - **指導ポイント：**アクセスログの各要素が何を意味するのかを丁寧に説明。正規表現は難易度が高いため、簡単な文字列分割 .split() から導入し、正規表現はオプションまたは応用として扱うことを推奨。エラー処理（形式が違う行をスキップするなど）の重要性にも触れる。
- **Part 2：データ加工と集計（ETL）**
  - **指導ポイント：**抽出したデータを DataFrame に入れるメリット（構造化、集計の容易さ）を強調。日時データの変換（pd.to_datetime）、不要な情報（画像ファイルへのアクセスなど）のフィルタリング、PV/UU の定義と集計方法を明確に説明。
- **Part 3：データマート作成と BI ツール①（Tableau）**
  - **指導ポイント：**「なぜこの列だけ CSV に出力するのか？」というデータマート設計の意図を説明。Tableau Public の基本的な操作（データ接続、ディメンション/メジャー、ドラッグ＆ドロップでのグラフ作成、ダッシュボード配置）をデモ中心に進める。
- **Part 4：BI ツール②（Power BI）と考察**
  - **指導ポイント：**Power BI Desktop での同様のダッシュボード作成手順を説明。Tableau との UI や操作感の違いを意識させる。完成したダッシュボードから読み取れるインサイト（よく見られるページ、アクセスが多い時間帯など）を学生に考えさせ、発表させる時間を設けると効果的。

ビッグデータ技術応用
実習 2　アクセスログ ETL と BI 可視化
教員用指導ガイド

§ 📋 このマニュアルについて

実習 2（Part 1〜4）を指導する教員・TA 向けのマニュアルです

内容：指導目標、よくあるトラブルと対処法、指導ポイント、進捗確認リスト、時間配分アドバイス

§ 🎯 全体的な指導方針

💡 基本姿勢

- **ETL の重要性を強調：**データ分析の前工程がいかに重要かを伝える
- **ツールの違いを意識させる：**Python, Tableau, Power BI それぞれの得意分野と使い分けを意識させる
- **正規表現は「使う」ことを目標に：**完璧な理解より、パターンをコピペして使えるレベルを目指す
- **BI ツール操作は「試行錯誤」を促す：**ドラッグ＆ドロップで色々試す中で発見があることを伝える

対象学生レベル

実習 1 を完了し、pandas の基本操作（読み込み、フィルタリング、groupby）に慣れていることを前提とします

§ 🔧 Part 1：アクセスログと ETL の基礎

よくあるトラブルと対処法

❌ トラブル 1-1：ログファイルの文字化け

症状

pd.read_csv() や open() で読み込んだ際に UnicodeDecodeError が発生する

原因

ファイルのエンコーディング（文字コード）と Python が想定しているエンコーディングが違う

対処法

1. read_csv() や open() に encoding='ISO-8859-1' (または 'latin-1') を指定するよう指示
2. それでもダメな場合は、他のエンコーディング（'shift-jis', 'cp932' など）を試す

予防

ワークブックに encoding='ISO-8859-1' を最初から記載しておく

❌ トラブル 1-2：正規表現がマッチしない（match = None）

症状

log_pattern.match(line) の結果が None になり、情報が抽出できない

原因

1. ログの形式が想定している Common Log Format と微妙に違う
2. 正規表現パターン自体に誤りがある（可能性は低い）
3. 行頭・行末に余計な空白がある

対処法

1. line.strip() で前後の空白を除去しているか確認
2. マッチしないログ行を具体的に表示させ、パターンと見比べてどこが違うか特定する
3. （高度）正規表現デバッガサイト（例: regex101.com）でパターンを検証する

指導のポイント

💡 **Part 1 指導上の工夫**

- ETL の「なぜ（Why）」を最初にしっかり説明する
- 正規表現は「魔法の杖」として紹介し、詳細な文法より「何ができるか」に焦点を当てる
- DataFrame に変換できた瞬間の「達成感」を共有する

✓ **Part 1 進捗確認**

- ログファイルを読み込めた
- 正規表現で主要情報（IP, time, url, status）を抽出できた
- 抽出結果を DataFrame に変換できた

§ 🧱 Part 2：データ型の整備とデータマート基礎

よくあるトラブルと対処法

❌ **トラブル 2-1：pd.to_datetime でエラー**

症状

ValueError: time data '...' does not match format '...' などが発生

原因

1. format=... の指定が、実際のログの形式と一致していない
2. ログファイル内に、想定外の形式の日時文字列が混入している

対処法

1. format 指定子（%d, %b, %Y など）が正しいか、ワークブックと見比べるよう指示
2. errors='coerce' オプションを付けて実行し、エラー箇所を特定後、原因調査（または無視

❌ **トラブル 2-2：.dt アクセサが使えない**

症状

AttributeError: Can only use .dt accessor with datetimelike values が発生

原因

対象の列が正しく日時型（datetime）に変換されていない（まだ object 型のまま）

対処法

1. 直前の pd.to_datetime の処理が成功しているか確認させる
2. df.info() で対象列の Dtype が datetime64 になっているか確認させる

指導のポイント

💡 **Part 2 指導上の工夫**

- 日時変換の format 指定子の重要性を強調する（実務で頻出）
- データマートの「目的志向」な考え方（分析に必要なものだけ選ぶ）を説明する
- ETL の最後の「L (Load)」として .to_csv() を位置づける

✓ **Part 2 進捗確認**

- timestamp 列が日時型になった
- 年・月・曜日・時間などの列が追加された
- データマート用の列を選択した DataFrame を作成できた
- access_log_mart.csv ファイルを保存できた

§ 📊 Part 3：Tableau Public によるダッシュボード作成

よくあるトラブルと対処法

❌ トラブル 3-1：Tableau Public がインストールできない/起動しない

症状

インストールエラー、起動時エラー、ログインできない

原因

1. PC のスペック不足（メモリ、OS バージョン）
2. ネットワーク接続の問題
3. アカウント作成時のメール認証未完了

対処法

1. Tableau Public のシステム要件を確認させる
2. 別のネットワーク環境（自宅など）で試すよう指示
3. アカウント作成プロセスを再確認させる
4. （最終手段）教員 PC でデモを見せ、操作は後で試してもらう

❌ トラブル 3-2：CSV ファイルを接続できない

症状

「テキストファイル」で CSV を選択してもエラーが出る、文字化けする

原因

1. CSV ファイル自体が破損している。
2. エンコーディングの問題（通常 Tableau は自動認識）

対処法

1. Part 2 の .to_csv() を再実行してファイルを作り直す
2. メモ帳などで CSV を開き、中身が正常か確認する
3. Tableau のデータ接続画面で「テキストファイルオプション」を確認させる（区切り文字など）

❌ トラブル 3-3：地図が表示されない/位置がずれる

症状

ip 列をダブルクリックしても地図が出ない、または明らかに違う場所に点が表示される

原因

1. ip 列の地理的役割が正しく設定されていない
2. Tableau が IP アドレスから位置情報を推定できない（プライベート IP、特殊な IP など）
3. ネットワーク環境による位置情報推定の制限

対処法

1. データソース画面で ip 列の地理的役割が「IP アドレス」になっているか再確認
2. サンプルデータの問題である可能性を伝え、気にせず進めるよう指示

3. （高度）緯度経度情報があればそれを使う方法もあることを補足

❌ トラブル 3-4：Tableau Public に保存できない

症状

「Tableau Public に保存」時にエラーが出る。

原因

1. Tableau Public アカウントでログインしていない。
2. ネットワーク接続が不安定。
3. ワークブック名に特殊文字が含まれている。

対処法

1. ログイン状態を確認させる
2. ネットワーク接続を確認し、再試行
3. シンプルなワークブック名（英数字）で保存してみる

指導のポイント

💡 Part 3 指導上の工夫

- Tableau の「ドラッグ＆ドロップ」操作を基本とし、直感的にグラフが作れる楽しさを体験させる
- 「ディメンションを列/行に置くと軸になり、メジャーを置くと値になる」という基本ルールを伝える
- 完成したダッシュボードで「フィルターとして使用」を試し、インタラクティブ性を実感させる
- Public 版の制約（公開必須）を再度注意喚起する

✓ Part 3 進捗確認

- CSV を Tableau に接続できた
- 時間帯別（折れ線）、曜日別（棒）、地図グラフを作成できた
- 3 つのグラフを配置したダッシュボードを作成できた
- インタラクション（フィルター）を設定できた
- Tableau Public に保存・公開できた

§ 🖥️ Part 4：Power BI Desktop によるダッシュボード作成

よくあるトラブルと対処法

❌ トラブル 4-1：Power BI Desktop がインストールできない

症状

インストール要件を満たさない、エラーが出る

原因

1. Windows OS でない（Mac や Linux）
2. OS のバージョンが古い、.NET Framework が不足

対処法

1. Mac ユーザーには仮想環境などの代替策を案内（ワークブック記載）
2. システム要件を確認させ、OS アップデートなどを促す
3. （最終手段）Part 3 (Tableau) の提出をもって評価対象とする

❌ トラブル 4-2：Power Query でデータ型を変更できない

症状

データ型アイコンをクリックしても変更できない、エラーが出る

原因

1. 変換できない値（例：数値列に文字列）が含まれている
2. 元のデータ読み込みステップに問題がある

対処法

1. エラーメッセージを確認し、問題のある値や行を特定するよう指示
2. Power Query 内で問題のある値を置換・削除するステップを追加する
3. 一度「閉じて適用」せず、データソースから再読み込みしてみる

❌ トラブル 4-3：曜日の並び順が正しくならない

症状

データビューで「並べ替えの基準列」を設定しても、レポートビューのグラフで反映されない

原因

1. 基準列（dayofweek）が数値として正しく認識されていない
2. レポートビューのグラフ設定で、別の並び替えルールが優先されている

対処法

1. データビューで dayofweek 列のデータ型が「整数」になっているか確認
2. レポートビューでグラフを選択し、「視覚化」ペイン右上の「...」から「軸の並べ替え」で dayname を選択し、昇順/降順を確認

❌ トラブル 4-4：地図が表示されない

症状

マップ視覚化を追加し、「場所」に ip 列を入れても地図が出ない

原因

1. データビューで ip 列の「データ カテゴリ」が「IP アドレス」に設定されていない
2. Power BI の地図機能が IP アドレスを認識できない（Tableau と同様）
3. ネットワーク/ファイアウォールの設定で地図サービスへのアクセスがブロックされている

対処法

1. データビューでのカテゴリ設定を再確認
2. サンプルデータの問題の可能性を伝え、気にせず進めるよう指示
3. （組織環境の場合）IT 部門にネットワーク設定を確認依頼

指導のポイント

💡 Part 4 指導上の工夫

- Power BI の画面構成（フィールド、視覚化、キャンバス）を最初にしっかり説明する
- データ加工（Power Query）とレポート作成（レポートビュー）が別画面であることを意識させる
- Tableau との操作感の違い（シェルフ vs ペイン、フィルター設定など）を比較しながら説明すると理解が深まる
- 「保存」がローカルの.pbix ファイルであることを強調する

✓ Part 4 進捗確認

- CSV を Power BI に取得できた
- Power Query でデータ型を修正できた
- 時間帯別（折れ線）、曜日別（棒）、地図グラフを作成できた
- 3 つのグラフを配置したレポートを作成できた
- グラフ間のインタラクションを確認できた
- .pbix ファイルとして保存できた

Part	内容	ポイント
Part 1	ETL 基礎、正規表現	正規表現で詰まる学生が多い。時間を多めに確保
Part 2	データ型整備、データマート	日時変換の format 指定、データマートの概念理解に時間をかける
Part 3	Tableau Public	ツールのインストール確認を事前に行う。操作に慣れる時間が必要
Part 4	Power BI Desktop	Windows 環境必須。Tableau との比較を意識させる

進捗調整：

- 正規表現や BI ツールに慣れていない学生が多い場合、各 Part の時間を延長するか、一部のタスク（例：URL クリーニング）を省略する
- 逆に進捗が早い場合は、積極的に発展課題を提示する

ビッグデータ技術応用
実習３　Spark による大規模データ処理
教員用導入ガイド

§ 🎯 実習の概要と指導目標

本実習は、PC のメモリに収まらない大規模データ（数 GB レベル）を対象とし、分散処理フレームワーク Apache Spark を用いたデータ処理の基本を体験させることを目的とします。

基礎教材との関連：

- 第 5 章：列指向ストレージ（Parquet 形式でのデータ読み書き）
- 第 7 章：大規模分散処理（Spark の概念、分散処理の必要性）

指導目標：

- 学生が単一マシン処理（pandas）の限界を認識し、分散処理（Spark）の必要性と効果を説明できるようになる
- 基本的な PySpark コード（DataFrame API 中心）を読み書きし、簡単な集計処理を実行できるようになる
- Spark 実行環境（Colab/Docker）のセットアップ手順を理解し、実行できるようになる

§ 💻 必要な環境とツール

📦 実行環境の選択肢と指導上の注意

🅰 オプション 1：Google Colaboratory（推奨）

指導のポイント：

- 最も環境構築トラブルが少ないため、授業進行がスムーズ
- 無料枠でのリソース制限（メモリ、ディスク、実行時間）があるため、扱うデータサイズの上限（例：5GB 程度）や処理時間に配慮が必要。Part 1 で生成するデータ量を調整する
- Spark のセットアップコマンド（!pip install pyspark 等）をワークブックに明記し、実行させる

🅱 オプション 2：ローカル環境（Docker）

指導のポイント：

- Docker Desktop のインストールと基本操作（イメージ取得、コンテナ起動）を事前に指導するか、詳細な手順書を提供する必要がある
- 推奨イメージ（例: jupyter/pyspark-notebook）を提示し、ポートマッピングやボリュームマウントの設定方法を解説する
- PC スペック（特にメモリ）によっては、Docker コンテナの起動・動作が不安定になる可能性があるため、注意喚起が必要
- 環境構築に時間がかかるため、授業時間外での準備を推奨。発展課題として位置づけるのが現実的

💡 環境選択の推奨

特別な理由がない限り、**Google Colaboratory を標準**とし、Docker 環境は意欲のある学生向けのオプションまたは発展課題として案内するのが、授業運営上は望ましいです。

§ 📂 使用するデータセット

📊 擬似センサーデータの生成

実習では、Part 1 のワークブック内で Python コードを実行し、大規模な擬似センサーデータ（CSV）を生成します。

💡 データ生成に関する指導ポイント

- **生成する行数の調整：** 学生の環境（特に Colab 無料枠）に合わせて、生成するデータサイズ（行数）を調整してください。数千万行〜1 億行（ファイルサイズで 2GB〜5GB 程度）が pandas では厳しく、Spark の効果を体感しやすい目安です。
- **生成時間の告知：** データ生成には時間がかかる（数分〜十分以上）ことを事前に伝え、その間に分散処理の概念などを解説する時間を設けると良いでしょう。
- **ディスク容量の確認：** Colab 無料枠ではディスク容量にも制限があります。生成前に空き容量を確認するよう促してください。`!df -h` コマンドで確認できます。
- **Parquet 変換：** 生成した CSV を Parquet 形式に変換するステップ（Part 2）の重要性（ファイルサイズ削減、読み込み速度向上）を強調してください。

代替データソース（参考）

もし、事前に大規模データを用意したい場合、以下の公開データセットなども検討可能です（ただし、前処理が必要になる場合があります）。

- **NYC Taxi Trip Data (TLC):** 数 GB〜数十 GB/月。CSV 形式
- **Wikipedia Page Views:** Parquet 形式で公開

§ 📖 Part 別の学習内容と指導ポイント

Part 1：分散処理の必要性と Spark 環境

- **内容：** 大規模 CSV データを pandas で読み込もうとしてメモリ不足エラーを体験させる。分散処理の基本概念を解説。Colab（or Docker）で Spark セッションを開始する。
- **指導ポイント：** pandas がなぜ遅い/失敗するのか（シングルコア、メモリ限界）を体感させることが重要。「メモリに乗らないデータをどう扱うか？」という問いから Spark 導入へ繋げる。

Part 2：Spark DataFrame の基本操作

- **内容：** Spark による CSV/Parquet の読み込み（`spark.read.csv/parquet`）。DataFrame の基本操作（`.show()`, `.printSchema()`, `.count()`）。簡単な集計（`.groupBy().agg()`）とフィルタリング（`.filter()` / `.where()`）。
- **指導ポイント：** pandas の DataFrame 操作との類似点・相違点を意識させる（API は似ているが、遅延評価される点が違う）。Parquet 読み込みの速さを CSV と比較させる。

Part 3：パフォーマンス比較と考察

- **内容：** 同じ集計処理（例：センサーごとの平均値算出）を pandas と Spark で実行し、処理時間（%%time マジックコマンド等）を比較する［cite: 436］。なぜ Spark が速いのか（分散処理、メモリ管理）を考察させる。
- **指導ポイント：** 単純な時間比較だけでなく、「データ量がさらに増えたらどうなるか？」「CPU コア数が増えたらどうなるか？」といったスケーラビリティの観点を考えさせることが重要。

Part 4：データ集計と可視化（応用）

- **内容：** より実践的な集計（例：時間窓での集計、センサー異常値の検出）。Spark で集計した結果を pandas DataFrame に変換（.toPandas()）し、matplotlib/seaborn で可視化する。
- **指導ポイント：** Spark は大規模データの「処理」が得意だが、最終的な「可視化」は pandas+matplotlib/seaborn で行うことが多い、という連携パターンを示す。.toPandas() は結果がメモリに乗るサイズになってから使う、という注意点を伝える。

§ ⚠️ よくあるトラブルと対処法

❌ トラブル：Colab で Spark セッションが開始できない/遅い

症状

SparkSession.builder.getOrCreate() がエラーになる、または非常に時間がかかる

原因

1. pyspark のインストールが不完全
2. Colab のランタイムリソース（メモリ）が不足している
3. 他のユーザーが多く利用している時間帯

対処法

1. Colab の「ランタイム」→「ランタイムの再起動」を試す
2. !pip install pyspark が正しく実行されたか確認
3. 処理するデータサイズを小さくしてみる
4. 時間帯を変えて試してみる

❌ トラブル：Spark の処理中にメモリ不足エラー

症状

java.lang.OutOfMemoryError, ExecutorLostFailure などが発生

原因

1. 処理対象データに対して、Spark に割り当てられたメモリ（Executor Memory）が不足している
2. 非効率な処理（例：巨大な DataFrame 全体をメモリに読み込もうとする操作）

対処法

1. （Colab Pro など）よりメモリの多い環境を利用する
2. 処理データ量を減らす
3. Spark のメモリ設定を変更する（ローカル/Docker 環境の場合）
4. コードを見直し、より効率的な処理（フィルタリングを先に行うなど）に書き換える

❌ トラブル：.toPandas() でメモリ不足エラー

症状

Spark での集計は成功したが、結果を pandas に変換しようとするとエラーになる

原因

集計結果のデータサイズが、まだ pandas で扱えるメモリサイズを超えている

対処法

1. さらに集計を進めるか、.limit() で一部のデータだけを pandas に変換して可視化する
2. Spark の可視化ライブラリ（Koalas など、ただし設定が必要）の利用を検討する（発展）

ビッグデータ技術応用
実習３　Spark による大規模データ処理
教員用指導ガイド

§ 📋 このマニュアルについて

実習 3（Part 1〜4）を指導する教員・TA 向けのマニュアルです。
Spark の概念説明、環境構築、パフォーマンス比較など、学生がつまずきやすいポイントを中心に、指導上の注意点とトラブルシューティングを提供します。

§ 🎯 全体的な指導方針

💡 基本姿勢

- **「なぜ Spark か？」を体感させる：** Part 1 の pandas 限界体験と Part 3 の比較実験を重視する
- **概念と実践のバランス：** 分散処理や遅延評価の概念を図解などで説明しつつ、まずはコードを実行して「動く」体験を優先する
- **API は DataFrame 中心に：** RDD API は紹介程度に留め、pandas 経験者が親しみやすい DataFrame API を中心に指導する
- **エラーを恐れない姿勢：** 大規模データ処理ではエラーはつきもの。「エラーメッセージを読む」「原因を推測する」「調べて解決する」プロセスを指導する

対象学生レベル

実習 1, 2 を完了し、Python/pandas の基本、ETL、BI ツールの概要を理解していることを前提とします

§ 🔧 Part 1：分散処理の必要性と Spark 環境

よくあるトラブルと対処法

❌ トラブル 1-1：Pandas でメモリ不足にならない

症状

Task 1-2 で巨大 DataFrame 生成が成功してしまう

原因

1. 生成行数が Colab 環境のメモリに対して少なすぎる
2. Colab Pro など、メモリの多い環境を使っている

対処法

1. num_rows_pandas を増やして再実行させる（2 倍、3 倍…と試す）
2. 成功した場合でも、「データがもっと大きくなれば必ず限界が来る」ことを強調する
3. Part 3 の集計処理で時間がかかることを示唆する

❌ トラブル 1-2：PySpark インストールでエラー

症状

!pip install pyspark がエラーで失敗する

原因

1. Colab のネットワーク接続が一時的に不安定
2. Colab の環境自体に問題がある（稀）

対処法

1. 時間を置いて再実行させる
2. Colab の「ランタイム」→「ランタイムの再起動」を試す
3. 「ランタイム」→「ランタイムのタイプを変更」でリセットしてみる

❌ トラブル 1-3：SparkSession 作成でエラー

症状

SparkSession.builder... で Java 関連のエラーなどが発生

原因

1. PySpark インストールが不完全
2. Colab 環境の一時的な不具合
3. （ローカルの場合）JAVA_HOME 設定などが正しくない

対処法

1. PySpark の再インストール（!pip uninstall pyspark -y してから !pip install pyspark）
2. ランタイムの再起動。
3. （ローカルの場合）Java のインストールと環境変数設定を確認させる

❌ トラブル 1-4：データ生成が異常に遅い/終わらない/ディスク不足

症状

Task 4-2 の CSV データ生成が終わらない、またはディスク容量不足エラー

原因

1. 生成行数が環境に対して多すぎる
2. Colab のディスク空き容量が不足している

対処法

1. NUM_ROWS_TO_GENERATE の値を小さくして再実行させる（例：1000 万行）
2. Colab の「ランタイム」→「ランタイムの接続解除と削除」でディスクをクリアしてから再試行
3. 事前に !df -h . で空き容量を確認するよう指示

指導のポイント

💡 Part 1 指導上の工夫

- Pandas の限界体験は「失敗」を恐れずにやらせる。エラーこそが学びの動機付け
- 分散処理の概念は、身近な例（グループワーク、大掃除の分担など）で説明する
- Colab での Spark セットアップが簡単であることを強調し、心理的ハードルを下げる
- データ生成の待ち時間を利用して、Spark アーキテクチャや用語（Driver/Executor など）の補足説明を行う

✓ Part 1 進捗確認

- Pandas の限界（MemoryError or 長時間）を体験した
- Colab で PySpark をインストールし、SparkSession を作成できた
- 指定した行数の大規模 CSV データを生成できた

§ ⚙️ Part 2：Spark DataFrame の基本操作

よくあるトラブルと対処法

❌ トラブル 2-1：CSV/Parquet 読み込みが非常に遅い/失敗する

症状

spark.read.csv/parquet() 後の Action (.show(), .count()) が異常に遅い、またはエラーになる

原因

1. データサイズに対して Colab のリソースが不足
2. inferSchema=True (CSV) が大規模データに対して非効率
3. ファイルパスが間違っている

対処法

1. データサイズを小さくして試す（Part 1 の生成行数を減らす）
2. CSV 読み込み時にスキーマを明示的に定義する（発展課題）
3. ファイルパスが正しいか !ls コマンド等で確認
4. ランタイム再起動

❌ トラブル 2-2：DataFrame 操作の API が分からない

症状

Pandas と同じ感覚で書いたらエラーになる（例: df['col'] > 30）

原因

Spark DataFrame API の構文に慣れていない

対処法

1. 基本的な操作（select, filter, groupBy/agg）の構文をワークブックや公式ドキュメントで確認させる
2. filter() では df.col > 30 や F.col('col') > 30 のように書くことを教える
3. 集計関数は pyspark.sql.functions (F) からインポートして使うことを再確認

指導のポイント

💡 Part 2 指導上の工夫

- **遅延評価の再強調：** Transformation だけでは何も起こらず、Action で初めて計算が走ることを、コード実行のタイミングで繰り返し説明する
- **Pandas API との比較：** 似ている点（select, groupBy など）と違う点（条件指定の書き方、集計関数の使い方）を明確にする
- **Parquet の優位性強調：** CSV と Parquet の読み込み速度の違い（特に .show() 実行時）を体感させ、ファイルサイズの違いも確認させる

✓ Part 2 進捗確認

- Spark で CSV/Parquet を読み込めた
- スキーマ確認、行数カウントを実行できた
- Select, Filter, GroupBy/Agg の基本操作を実行できた
- Parquet 形式で保存・再読み込みできた

§ ⚖️ Part 3：パフォーマンス比較と考察

よくあるトラブルと対処法

❌ トラブル 3-1：%%time が機能しない/使い方が分からない

症状

実行時間が表示されない、エラーになる

原因

1. %%time がセルの先頭に書かれていない
2. Jupyter/Colab 環境以外で実行しようとしている

対処法

1. 必ずセルの**一番最初**に %%time を書くよう指示
2. これが Jupyter/Colab のマジックコマンドであることを説明
3. 通常の Python スクリプトでは time モジュールを使う方法があることを補足

❌ トラブル 3-2：比較結果の解釈が難しい

症状

Pandas がエラーになり比較できない、または Spark との差が思ったほど出ない。

原因

1. データサイズが小さすぎる（Spark の分散オーバーヘッドの方が大きい）
2. Colab 環境のリソース状況によるばらつき
3. 比較している処理内容が単純すぎる

対処法

1. Pandas がエラーになること自体が「限界」を示す重要な結果であることを説明
2. 差が出にくい場合は、データサイズを大きくする、またはより複雑な処理（例：複数回の GroupBy/Join）で比較する必要があることを示唆（発展課題）
3. 重要なのはオーダー（桁）の違いであり、細かい秒数は環境依存であることを伝える

指導のポイント

💡 Part 3 指導上の工夫

- **比較実験の目的明確化：** Spark の「速さ」と「スケーラビリティ」を体感させることが目的
- **時間の計測方法：** %%time の見方（Wall time）を説明
- **考察を促す質問：** 「なぜ Spark は速い？」「もしデータが 10 倍になったら？」など、結果の裏にあるメカニズムを考えさせる。分散処理、遅延評価の概念と結びつけさせる

✓ Part 3 進捗確認

- Pandas での処理時間（またはエラー）を記録できた
- Spark での処理時間を記録できた
- 両者のパフォーマンスの違いを認識できた
- Spark が高速な理由を考察・記述できた

§ 📊 Part 4：データ集計と可視化（応用）

よくあるトラブルと対処法

❌ トラブル 4-1：.toPandas() でメモリ不足エラー

症状

Spark での集計は成功したが、.toPandas() で MemoryError

原因

集計結果のサイズが、まだ Driver（Colab）のメモリに収まるほど小さくなっていなかった

対処法

1. .toPandas() の前に、さらに集計を進めるか、.limit() で行数を制限してから変換するよう指示
2. なぜエラーになったか（全データが Driver に集まるため）のメカニズムを説明
3. .toPandas() は「最後の手段」であり、可能な限り Spark 内で処理を完結させるのが理想であることを補足

❌ トラブル 4-2：可視化コードがうまく動かない

症状

.toPandas() は成功したが、matplotlib/seaborn でグラフが描画できない、エラーが出る

原因

1. Pandas DataFrame の列名やデータ型が可視化コードと合っていない
2. 実習 1, 2 で学んだ可視化の基本を忘れている

対処法

1. hourly_avg_pd_df.info() や .head() で、変換後の Pandas DataFrame の構造を再確認させる。
2. 実習 1, 2 のワークブック（特に Part 3）の可視化コードを参考にするよう指示
3. エラーメッセージを読んで、どの部分で問題が起きているか特定させる

指導のポイント

💡 Part 4 指導上の工夫

- **Spark + Pandas/Vis 連携の意義：**「Spark で重い処理、Pandas/Vis で仕上げ」という実践的なワークフローを示す
- **.toPandas() の功罪：** 便利だがメモリリスクがあることを強く意識させる
- **応用集計の紹介：** ウィンドウ関数や Spark SQL は深入りせず、「Spark にはこんな高度な機能もある」と紹介する程度で良い
- **実習全体のまとめ：** 実習 1, 2, 3 を通して学んだことの繋がり（データ分析プロセス→ETL/BI→大規模処理）を振り返り、全体像を提示する

✓ Part 4 進捗確認

- Spark で応用的な集計を実行できた
- .toPandas() を使って結果を Pandas に変換できた
- 変換後の Pandas DataFrame を可視化できた
- Spark と Python エコシステム連携の流れを理解できた

人工知能（ＡＩ）技術応用

人工知能（AI）技術応用
実習１　機械学習で手書き数字を認識
導入編　教員用指導ガイド

§ 1. 実習の目的と位置づけ

🎯 メインテーマ：機械学習の基礎（分類問題）と評価の実践

AI 技術応用の最初のステップとして、機械学習の最も基本的なタスクである「画像分類」に取り組みます。7 万枚の手書き数字データ（MNIST）を使い、データの準備からアルゴリズムの実装、性能評価、そして実データでの検証まで、機械学習の一連のワークフローを体験します。

学生の到達目標

- Google Colab の基本操作を習得し、クラウド環境での AI 開発に慣れる。
- 機械学習における「訓練データ」と「テストデータ」の役割と分割の重要性を説明できる。
- 3 つの異なるアルゴリズム（決定木、k 近傍法、ロジスティック回帰）の特徴を理解し、実装できる。
- 正解率（Accuracy）や混同行列（Confusion Matrix）を用いて、モデルの性能を客観的に評価・比較できる。
- 自分自身の描いた手書き数字をテストすることで、画像の前処理（リサイズ、階調反転等）の重要性を体感する。

前提知識と対象

- **対象：** Python と機械学習の基本的な概念（学習と予測）を学び始めた学生。
- **難易度：** 初中級（コードはライブラリを活用するため、ロジックの理解に集中できます）。
- **前提知識：** Python の基礎的な文法、Google アカウントの所有。

§ 2. 実習の全体構成（全 4 Part）

全実習時間は、講義・実習・考察を含め、 **4〜5 コマ（200〜250 分）** を目安とします。

Part	テーマ	主な学習内容
1	環境準備とデータ確認	Google Colab の設定、MNIST データ読込、訓練/テストデータの分割、データの可視化。
2	3 つのアルゴリズム実装	決定木、k 近傍法、ロジスティック回帰の 3 モデルを実装し、一部のデータで予測結果を確認。
3	結果の比較と評価	全テストデータでの正解率算出、モデル間の性能比較、混同行列による誤分類傾向の分析。

Part	テーマ	主な学習内容
4	実データでの検証	自分で作成した画像データのアップロード、画像の前処理（リサイズ・グレースケール）、AI による判定。

§ 3. 指導上の工夫と重要ポイント

① 3 つのアルゴリズムを「身近な例」で説明する

各手法の数式を追うのではなく、直感的なイメージを持たせることで理解を促します。

アルゴリズム	教員用説明の例え	ポイント
決定木	「20 の質問ゲーム」	「線は曲がっているか？」等の Yes/No 判断の積み重ね
k 近傍法	「類は友を呼ぶ（近くの友達に聞く）」	特徴が似ている（距離が近い）データの多数決で決める
ロジスティック回帰	「確率で考える（降水確率と同じ）」	「これが 3 である確率は 80%」のように確信度で判断する

② 評価の「客観性」を意識させる

Part 2 では数件の結果を見るだけですが、Part 3 では「1 万件の統計」で評価します。

- 「たまたま数件当たった」のと「全体の 9 割以上が当たっている」ことの違いを強調してください。
- **混同行列の活用：** 「3 を 8 と間違えやすい」といった AI の"癖"を発見させ、なぜ人間でも見間違えるのかを議論させると効果的です。

③ 実データ特有の「壁」を体験させる

Part 4 では、用意されたデータセット（MNIST）と、自分たちの現実世界のデータの違いを学びます。

- **前処理の重要性：** 「そのままでは判定できない」ことを体験し、AI に合わせた形（サイズや色の濃さ）に加工する工程が、開発時間の大部分を占めることを伝えてください。
- **失敗からの学び：** 自分の書いた数字が認識されなかった場合、それを「失敗」とするのではなく、「なぜ MNIST のデータと違うと判断されたのか（ペンの太さ、背景の影など）」を考察させることが重要です。

§ 💡 教員へのアドバイス：

実習 1 は「動いた！」という成功体験を積ませるのに最適です。コードの細かなスペルミス（特に対象括弧など）で止まる学生が多いため、それらをサポートすることを推奨します。

人工知能（AI）技術応用
実習１　機械学習で手書き数字を認識
Part１：環境準備とデータ確認　教員用指導ガイド

§ 🎯 指導目標

この Part で学生が達成すべきこと：

- Google Colab の基本操作に慣れ、自分でノートブックを作成・実行できる
- MNIST データセットを読み込み、内容を理解できる
- 機械学習における訓練データとテストデータの役割を理解できる
- データの可視化を通じて、問題の性質を把握できる
- 次の Part でアルゴリズムを学ぶための基礎を固める

§ タスクごとの内容

項目	内容
導入・説明	Part1 の目的、全体の流れ説明
タスク 1-2（環境確認）	Google Colab アクセス、Python 動作確認
タスク 3-5（データ読込）	ライブラリ、MNIST 読込、データ分割
タスク 6-7（可視化）	画像表示、統計情報確認
考察問題	振り返り、理解度確認
質疑応答・まとめ	疑問点解消、次回予告

💡 **時間調整のポイント**

- 環境設定で時間がかかる学生が必ずいるため、余裕を持たせる
- データ読込みは初回のみ時間がかかる（ダウンロード）
- 考察問題は宿題にしてもよい

§ 💡 指導上の工夫とポイント

1. 導入時の工夫

✨ **効果的な導入方法**

- **デモを見せる：**最初に完成形（数字を認識する AI）を実演し、ゴールを示す

- **不安を取り除く**：「エラーが出ても大丈夫」「分からなかったら質問してね」と伝える
- **身近な例で説明**：「郵便番号の自動読取」など実用例を紹介

2. Google Colab 使用時の注意点

⚠️ **事前に確認すべきこと**

- 全員が Google アカウントを持っているか確認（前日までに）
- 学校のネットワークで Colab がブロックされていないか確認
- Chrome または Edge ブラウザの使用を推奨
- タブを閉じても作業は保存されることを説明

3. コード実行時の声かけ

🗣️ **効果的な声かけ例**

- 「実行ボタン（▶）を押したら、少し待ってね」
- 「エラーが出た人は手を挙げて。一緒に確認しましょう」
- 「周りの人と進捗を確認し合いましょう」
- 「結果が表示されたら、隣の人と見比べてみよう」

4. データ可視化時の指導

👁️ **観察を促す質問**

- 「どの数字が一番読みやすい？読みにくい？」
- 「人間でも間違えそうな数字はある？」
- 「なぜ AI に学習させる必要があるのか考えてみよう」

§ 🔧 よくあるトラブルと対処法

❌ **トラブル 1: Google Colab にアクセスできない**

症状：

- ログイン画面が表示されない
- 「アクセスが拒否されました」と表示される
- ページが真っ白のまま

原因：

- 学校のネットワークでブロックされている
- ブラウザのクッキーが無効
- ブラウザのバージョンが古い

✅ **対処法：**

1. **ブラウザを変更**：Chrome または Edge を試す
2. **シークレットモードで開く**：Ctrl+Shift+N（Chrome）
3. **クッキーを有効化**：設定 → プライバシーとセキュリティ
4. **IT 管理者に相談**：学校のファイアウォール設定を確認

🛡️ **予防策：**

- 授業前日に全員がアクセスできるか確認する
- アクセスできない場合の代替案を準備（ローカル Jupyter 等）

❌ **トラブル 2: コードを実行してもエラーが出る**

症状：

- 赤い文字でエラーメッセージが表示される
- 「NameError」「SyntaxError」などのエラー
- コードは実行されるが、期待した結果が出ない

原因：

- コピー&ペースト時に余計なスペースや改行が入った
- セルを順番通りに実行していない
- スペルミス（特に大文字・小文字の違い）
- 全角スペースが混入

✅ **対処法：**

1. **エラーメッセージを確認**：最後の行に原因のヒント
2. **セルを上から順に再実行**：ランタイム → すべてのセルを実行
3. **コードを再入力**：手打ちで入力し直す
4. **インデント確認**：全角スペースになっていないか
5. **比較確認**：正しいコードと見比べる

§ 💡 エラー別対処法：

- NameError: name 'X' is not defined → 前のセルを実行していない
- SyntaxError: invalid syntax → スペルミスまたは記号の間違い
- IndentationError → インデント（字下げ）の間違い

❌ **トラブル 3: データのダウンロードが遅い・失敗する**

症状：

- MNIST データ読込みで長時間待たされる
- 「タイムアウト」エラーが出る
- 読込みが途中で止まる

原因：

- インターネット接続が不安定
- 多数の学生が同時にダウンロード
- サーバー側の一時的な問題

✅ **対処法：**

1. **待つ**：初回は 3-5 分かかることもある（我慢強く待つ）
2. **再実行**：失敗したら少し時間をおいて再実行
3. **ランタイム再起動**：ランタイム → ランタイムを再起動
4. **時間をずらす**：全員が同時実行しないよう指示

🛡️ **予防策：**

- 事前に教員用アカウントでデータをダウンロードしておく
- 「少し時間がかかります」と事前アナウンス
- グループごとに時間差で実行させる

❌ **トラブル 4: 画像が表示されない**

症状：

- グラフ・画像が表示されない

- 真っ白なまま
- エラーは出ないが何も表示されない

原因：

- plt.show()を忘れている
- データが正しく読み込まれていない
- matplotlib の設定問題

✅ **対処法：**

1. **show()確認：**plt.show()が書かれているか
2. **データ確認：**前のセル（データ読込）を再実行
3. **セルを統合：**グラフ作成と表示を同じセルに

§ ✅進捗確認チェックリスト

以下の項目を確認し、全員が達成できているか巡回しながらチェックしましょう。

タスク 1-2：環境準備

Google Colab にアクセスできている
ノートブックが作成できている
Python コードが実行できている
結果が表示されている

タスク 3-5：データ準備

ライブラリが読み込めている
MNIST データが読み込めている
データ分割ができている
枚数（60000/10000）を確認できている

タスク 6-7：可視化

手書き数字画像が表示されている
ラベルと画像が一致している
統計グラフが表示されている
データの特徴を観察している

考察・まとめ

考察問題に取り組んでいる
振り返りを記入している
ノートブックを保存している

§ ❓よくある質問と回答例

Q1: なぜ 784 ピクセルなんですか？

A: 28×28 の画像だからです。28×28=784 ですね。画像は縦 28 個、横 28 個のマス目（ピクセル）でできていて、それを 1 列に並べると 784 個になります。

補足：実際に計算させてみるとより理解が深まります。

Q2: なぜ訓練データとテストデータを分けるんですか？

A: 良い質問ですね！例えば、問題集で勉強した後、同じ問題でテストしたら 100 点取れますよね？でも、それは本当の実力ではありません。AI も同じで、学習に使ったデータで テストすると良い結果が出てしまいます。だから、見たことがないデータでテストする必要があるんです。

キーワード：「過学習（オーバーフィッティング）」の概念につながる（Part 3 で詳しく学ぶ）

Q3: 読みにくい数字は AI も間違えますか？

A: その通りです！人間が読みにくい数字は、AI も間違えやすいです。例えば、「1」と「7」の書き方が似ている場合などです。Part 2 でいろいろなアルゴリズムを試して、どのくらい正確に認識できるか確認しましょう。

発展：実際にどの数字の組み合わせが間違えやすいか、Part 3 の混同行列で確認できることを予告

Q4: 6 万枚も必要なんですか？もっと少なくてもいいのでは？

A: 鋭い質問ですね！実は、データが多いほど AI の性能は良くなります。少ないデータでも学習はできますが、いろいろな書き方を覚えるには、たくさんの例が必要です。後でデータ数を減らして実験してみてもいいですよ（発展課題）。

補足：データ量と精度の関係は機械学習の重要なトピック

§ 🚀次の Part に向けて

教員が準備すべきこと：

- Part 1 で躓いた学生のリストアップとフォロー
- Part 2 で使う 3 つのアルゴリズムの復習
- 学生用のサンプルコードの動作確認
- 予想される所要時間の再確認

学生に伝えること：

- 次回は実際に機械学習モデルを作ります
- 今日できなかったところは次回前に確認しておこう
- 考察問題はしっかり考えてきてね
- ノートブックを保存し忘れないように

人工知能（AI）技術応用
実習１　機械学習で手書き数字を認識
Part２：３つのアルゴリズム実装　教員用指導ガイド

§ 📋 Part 2　授業概要

項目	内容
学習目標	3 つの機械学習アルゴリズムを実装し、それぞれの特徴を理解する
前提知識	Part 1 完了（データの読み込み、基本的な Python 操作）

§ 🎯 授業の進め方

1. **導入**

📌 **指導のポイント**

- Part 1 で準備したデータを使うことを確認
- 「今日は 3 つの異なる方法で AI を作る」ことを強調
- 料理に例える：「同じ材料でも、作り方が違うと違う料理になる」

2. **アルゴリズムの説明**

📌 **指導のポイント**

- **決定木**：「20 の質問ゲーム」に例える
  - 「この線は曲がっている？」「丸がある？」のように質問を繰り返す
  - フローチャートのイメージを見せると分かりやすい
- **k 近傍法**：「友達に聞く」に例える
  - 「近くにいる 5 人に聞いて、多数決で決める」
  - 「類は友を呼ぶ」の原理
- **ロジスティック回帰**：「確率で考える」に例える
  - 「これが 3 である確率は 80%」のように確率で判断
  - 天気予報の降水確率と同じ考え方

💡 **効果的な説明方法**

各アルゴリズムを説明する際は、PowerPoint で視覚的に示しながら、実生活の例を使うと理解が深まります。「みんなならどの方法を使う？」と問いかけると、考えるきっかけになります。

§ ⚠️ よくあるトラブルと対処法

🔴 トラブル 1：「エラーが出て実行できない」

症状

NameError: name 'X_train' is not defined

原因

Part 1 のコードが実行されていない、またはノートブックを再起動した

✅ 解決方法

1. ノートブックの上から順番にすべてのセルを実行する
2. 「ランタイム」→「すべてのセルを実行」でも可
3. Part 1 のデータ読み込みコードを再実行

🔴 トラブル 2：「学習に時間がかかりすぎる」

症状

決定木や k 近傍法の学習が数分以上かかる

原因

データサイズが大きい、またはパラメータ設定が不適切

✅ 解決方法

1. 正常な学習時間：
    - 決定木：1-3 秒
    - k 近傍法：1-2 秒（学習は速い）
    - ロジスティック回帰：30-60 秒
2. 時間がかかりすぎる場合は、ノートブックを再起動
3. GPU 使用を確認（「ランタイム」→「ランタイムのタイプを変更」）

🔴 トラブル 3：「ConvergenceWarning が出る」

症状

ロジスティック回帰で黄色い警告メッセージが表示される

✅ 解決方法

これは正常です。「もっと長く計算すればより正確になる」という意味で、動作には問題ありません。

学生には「警告は出ているが、結果は使える」と説明してください。

警告を消したい場合：

```
lr_model = LogisticRegression( max_iter=1000, # 100 から 1000 に増やす random_state=42 )
```

🔴 トラブル 4：「予測結果が全部同じ数字になる」

症状

すべての予測が「0」や「1」など同じ数字になる

原因

データの正規化が正しく行われていない

✅ 解決方法

1. Part 1 のデータ正規化コードを確認
2. 以下のコードを実行して確認：

```
print("X_train の範囲:", X_train.min(), "〜", X_train.max()) # 0.0 〜 1.0 の範囲になっている
```

べき

3. 範囲が 0-255 の場合は、再度正規化：

X_train = X_train / 255.0 X_test = X_test / 255.0

🔴 **トラブル 5：「3 つのモデルの正解率が全く同じ」**

症状

決定木、k 近傍法、ロジスティック回帰の正解率が完全に一致する

原因

同じモデルを 3 回使っている可能性

✅ **解決方法**

1. 各セルで正しいモデルをインポートしているか確認
2. 変数名が正しいか確認（dt_model, knn_model, lr_model）
3. 正常な場合の目安：
    - 決定木：85-90%
    - k 近傍法：95-97%
    - ロジスティック回帰：92-93%

§ 💬 学生からのよくある質問と回答例

Q1：なぜ 3 つもアルゴリズムを試すのですか？

A：どのアルゴリズムが一番良いかは、データによって変わります。料理と同じで、魚には焼く、煮る、揚げるなど色々な調理法があり、その日の気分や目的で選びますよね。機械学習も同じで、複数試して比較することが大切です。

Q2：k 近傍法の「k」って何ですか？

A：「何人に聞くか」を決める数字です。k=5 なら「近くの 5 人に聞いて多数決」、k=10 なら「近くの 10 人に聞いて多数決」という意味です。多すぎても少なすぎても良くないので、適切な値を選ぶ必要があります。

Q3：決定木の「深さ」とは何ですか？

A：「何回質問するか」を決める数字です。max_depth=10 なら「最大 10 回質問できる」という意味です。深くすると細かく分類できますが、深すぎると「暗記」になってしまい、新しいデータでうまく働きません。

Q4：なぜロジスティック回帰だけ学習に時間がかかるのですか？

A：ロジスティック回帰は「計算を繰り返して少しずつ正解に近づく」方法だからです。決定木は「ルールを作るだけ」、k 近傍法は「データを覚えるだけ」なので速いですが、ロジスティック回帰は「最適な確率を計算する」ため時間がかかります。

Q5：確率が表示されるのはロジスティック回帰だけですか？

A：いいえ、実は決定木や k 近傍法でも確率は計算できます（predict_proba()メソッドを使う）。ただし、ロジスティック回帰は「確率を計算すること」が目的のアルゴリズムなので、確率の精度が高いと言われています。

§ ⚡ 指導上の注意点

1. **コードの暗記は不要と伝える**

この実習の目的は「プログラミング能力」ではなく「機械学習の考え方の理解」です。コードはコピペで進めて構わないことを明確に伝えましょう。

2. **「なぜ？」を考えさせる**

結果が出たら「なぜこうなったと思う？」と問いかけましょう。正解率の違い、学習時間の違いなど、観察から考えることを促します。

3. **エラーは学びのチャンス**

エラーが出たら、すぐに答えを教えるのではなく「どこが原因だと思う？」と考えさせる時間を取りましょう。ただし、5 分以上悩んでいたらヒントを出します。

4. **ペースの違いに配慮**

早く終わった学生には発展課題を案内し、遅れている学生にはペアを組ませるなど、個別対応を心がけましょう。

5. **成功体験を大切に**

どんな小さな成功でも褒めましょう。「コードが動いた」「エラーを自分で解決できた」など、肯定的なフィードバックが学習意欲を高めます。

§ ✅ 授業のまとめ方

授業の最後には、以下の点を確認しましょう：

1. **3 つのアルゴリズムの名前を覚えた**
2. **それぞれの特徴（質問型、多数決型、確率型）を理解した**
3. **同じデータでも、アルゴリズムによって結果が違うことを体験した**
4. **次回は性能を詳しく評価することを予告**

💡 **効果的なまとめ方**

「今日は 3 つの AI を作りました。どれが一番良かったですか？」と挙手で聞くと、学生の理解度が分かります。また「次回はもっと詳しく比較します」と予告することで、次の授業への期待感を高めます。

人工知能（AI）技術応用
実習１　機械学習で手書き数字を認識
Part３：結果の比較と評価　教員用指導ガイド

§ 📋 Part 3 授業概要

項目	内容
学習目標	モデルの性能を定量的に評価し、最適なアルゴリズムを選定できる
前提知識	Part 2 完了（3 つのモデル実装済み）

§ 📌 Part 3 の重要ポイント

- Part 2 では「最初の 10 件」だけ見たが、Part 3 では「全 10,000 件」で評価
- 正解率という「数値」で客観的に比較
- 混同行列で「どの数字を間違えやすいか」を分析
- データサイエンスの基本（評価指標、可視化）を学ぶ

§ ⚠️ よくあるトラブルと対処法

🔴 **トラブル 1：「正解率が異常に低い（50%以下）」**

原因と対処

✅ **確認ポイント：**

1. Part 2 のモデルが正しく学習されているか
2. テストデータ（X_test, y_test）が正しく準備されているか
3. データの正規化が正しく行われているか

正常な正解率の目安：

- 決定木：85-90%
- k 近傍法：95-97%
- ロジスティック回帰：92-93%

🔴 **トラブル 2：「混同行列が表示されない」**

原因

seaborn ライブラリがインストールされていない、または予測結果がない

✅ **解決方法：**

1. seaborn のインストール確認：!pip install seaborn
2. 予測が実行されているか確認（Part 2 のコード）
3. エラーメッセージを読んで原因を特定

🔴 トラブル3：「グラフが小さくて見えない」

✅ 解決方法：

`plt.figure(figsize=(10, 6))`の数値を大きくする（例：(14, 8)）

§ 💬 学生からのよくある質問

Q1：正解率90%は良いのですか？

A：はい、とても良いです！10回中9回正解なので、人間と同等かそれ以上です。ただし、用途によって求められる精度は違います。医療診断なら99%以上必要ですが、商品レコメンドなら80%でも実用的です。

Q2：なぜk近傍法が一番正解率が高いのですか？

A：MNISTデータは「似た画像は似た数字」という性質が強いからです。k近傍法は「似ているものを探す」ことが得意なので、このデータと相性が良いのです。

Q3：混同行列の見方が分かりません

A：縦軸が「本当の数字」、横軸が「予測した数字」です。対角線（左上から右下）が正解で、それ以外が間違いです。数字が大きいほど、そのパターンで間違えやすいということです。

§ ⚡ 指導上の注意点

1. **数値の意味を理解させる**

「90%」という数字だけでなく、「100枚中90枚正解」のように具体的に考えさせましょう。

2. **混同行列は難しい**

全員が完璧に理解する必要はありません。「どの数字のペアで間違えやすいか」が分かればOKです。

3. **最適なモデル選びを考えさせる**

「一番良いモデルはどれ？」だけでなく、「なぜそれを選ぶ？」を考えさせることが重要です。

人工知能（AI）技術応用
実習１　機械学習で手書き数字を認識
Part４ ：実データでの検証　教員用指導ガイド

§ 📋 Part 4 授業概要

項目	内容
学習目標	自分の手書き数字で AI をテストし、実習全体を振り返る
前提知識	Part 1-3 完了（モデル作成と評価）

§ 📌 Part 4 の重要ポイント

- 実データ（自分の手書き数字）でテストする実践的な体験
- 画像の前処理（リサイズ、グレースケール変換）の重要性
- 実習全体を振り返り、学びをまとめる
- 機械学習の実用性と限界を体感する

§ ⚠️ よくあるトラブルと対処法

🔴 **トラブル 1：「画像がアップロードできない」**

✅ **解決方法：**

1. ファイルサイズが大きすぎる場合：スマホで圧縮してから再アップロード
2. ファイル形式の問題：jpg, png, webp 形式に変換
3. Colab の左サイドバーから直接ドラッグ&ドロップ

🔴 **トラブル 2：「前処理後の画像が真っ黒/真っ白」**

✅ **原因：** 反転処理の問題、または撮影時の明るさの問題

✅ **解決方法：**

- 反転コード（255 - img_array）を削除または調整
- 撮影をやり直す（明るい場所で、影なく）
- 元画像を確認して問題を特定

🔴 **トラブル 3：「予測が全然当たらない」**

✅ **原因：**

- 数字の書き方が MNIST と大きく異なる
- 前処理が正しく行われていない
- 画像が小さすぎる/大きすぎる

✅ **対処：**

1. MNIST のような書き方で再度書く（太めのペン、はっきりと）
2. 前処理コードを確認
3. 「うまくいかないこともある」ことを学ぶ機会にする

§ 💬 学生からのよくある質問

Q1：なぜ自分の数字だと認識率が下がるのですか？

A：MNIST データは特定の書き方で作られているからです。AI は「学習したデータに似たもの」を認識するのが得意ですが、全く違う書き方だと苦手です。これは人間でも同じですよね。

Q2：どうすれば認識率を上げられますか？

A：3 つの方法があります：

1）書き方を MNIST に近づける（太めに、はっきりと）

2）自分の書き方のデータも学習に加える

3）もっと高度なアルゴリズム（ディープラーニング）を使う

Q3：実社会ではどう使われていますか？

A：郵便番号の自動読み取り、銀行の小切手処理、スーパーのレジでの商品認識などで使われています。ただし、実用では 99%以上の精度が求められるため、より高度なシステムを使っています。

§ ⚡ 指導上の注意点

1. 失敗も学びのチャンス

自分の数字がうまく認識されないことは、むしろ良い学習機会です。「なぜだめだったのか」を考えさせましょう。

2. 実用との違いを理解させる

今回の実習は「学習用」であり、実社会のシステムはもっと複雑で高精度であることを伝えましょう。

3. 達成感を大切に

4 つの Part を完走したことを称賛し、自信を持たせましょう。これが次の学習への動機づけになります。

4. 次のステップを示す

発展課題や、次に学ぶべきこと（ディープラーニング、他のデータセットなど）を紹介し、学習意欲を維持させましょう。

§ ✅ 最終チェックリスト

授業終了前に、以下を確認してください：

- ☐ 全員が Part 1-4 を完了している
- ☐ ワークブックがすべて記入されている
- ☐ 考察問題に答えている
- ☐ 最終レポートを作成している
- ☐ Colab ノートブックを保存している
- ☐ 提出方法を理解している

人工知能（AI）技術応用
実習 2 画像認識 AI で犬と猫を分類
導入編 教員用指導ガイド

§ 1. 実習の目的と位置づけ

🎯 **メインテーマ：ディープラーニング（CNN）と転移学習の実践**

実習 1（機械学習の基礎）から一歩進み、現代 AI の主流技術である「ディープラーニング」を用いて、複雑な画像データ（カラー写真）を扱います。ゼロからのモデル構築と、既存の賢いモデルを利用する「転移学習」の両方を体験し、その性能差を実感させます。

学生の到達目標

- 深層学習の基本モデルである CNN（畳み込みニューラルネットワーク）の仕組みを説明できる。
- 学習曲線を見て「過学習」を発見し、ドロップアウト等の対策を実装できる。
- 「転移学習」を活用し、短時間で高精度な AI を構築できる。

前提知識と対象

- **対象**：実習 1 を完了し、Python と機械学習の基本フロー（学習→予測）を理解している学生。
- **難易度**：高（概念的な難しさはありますが、コードはライブラリを活用するため記述量は適度です）。

§ 2. 実習の全体構成（全 4 Part）

全実習時間は、講義・実習・考察を含め、**4〜5 コマ（200〜250 分）**を目安とします。

Part	テーマ	主な学習内容
1	**環境準備と データ前処理**	• GPU 環境の設定（必須） • 画像データの構造（RGB/ピクセル）理解 • データ拡張（Data Augmentation）の実装
2	**CNN 構築と 過学習の発見**	• Keras による CNN モデル設計（畳み込み・プーリング） • モデルの学習実行 • 学習曲線から「過学習」を発見する
3	**モデルの 評価と改善**	• テストデータでの客観的評価 • 誤分類画像の分析（AI の弱点探し） • ドロップアウトによる過学習対策
4	**転移学習と**	• VGG16 を用いた転移学習の実装

Part	テーマ	主な学習内容
	実データテスト	• 自作モデルとの精度比較 • 学生自身の画像での最終テスト

§ 3. 指導上の工夫と重要ポイント

① CNN の仕組みを「人間の視覚」で説明する

CNN の構造は直感的に理解しにくい部分です。以下の対比表を使って説明すると効果的です。

AI の処理（CNN）	機能・役割	人間の視覚・脳の対応
畳み込み層 (Convolution)	画像からエッジ、線、色などの**特徴を抽出**するフィルタ。	目が「ここは縦線」「ここは赤い」と細部を捉える働き。
プーリング層 (Pooling)	画像のズレを許容し、情報を**圧縮**する。	対象が少し動いても「同じもの」と認識する脳の補正。
全結合層 (Fully Connected)	抽出された特徴から、最終的な**分類**を行う。	脳が「耳が尖っていてヒゲがあるから...猫だ！」と結論づける働き。

💡 **アクティビティの提案**実際に画像を加工アプリなどで「ぼかし」たり「輪郭抽出」したりするデモを見せると、畳み込み層のイメージが湧きやすくなります。

② 「過学習」を体験させる重要性

Part 2 では必ず過学習が発生します。これを「失敗」と捉えさせず、**「AI 開発における最大の敵を見つけた」**とポジティブに指導してください。

- **例え話：**「過去問（訓練データ）だけ丸暗記して、本番のテスト（検証データ）で点数が取れない学生の状態」

③ 「転移学習」の実用性

Part 4 の転移学習では、劇的に精度が向上します。「ゼロから頑張る」ことと、「先人の知恵（VGG16）を借りる」ことの効率の違いを強調し、実社会では後者が主流であることを伝えてください。

§ 4. よくあるトラブルと対処法

⚠️ **最重要：GPU の設定確認**

実習 2 は計算負荷が高いため、開始時に必ず Google Colab の「ランタイム」＞「ランタイムのタイプ

を変更」で GPU が選択されているか確認させてください。CPU のままでは学習が終わりません。

症状・質問	原因と対処法
エラー：RuntimeError: GPU not found	GPU ランタイムが有効になっていません。設定を変更し、再接続してください。
学習が進まない / 極端に遅い	GPU が使われていないか、バッチサイズが大きすぎる可能性があります。GPU 確認後、それでも遅ければバッチサイズを小さく（例: 32→16）してみてください。
精度が上がらない（Part 2）	正常です。単純な CNN をゼロから学習させるのは難しいため、Part 3 の改善や Part 4 の転移学習への伏線として利用してください。
実習 1 の決定木ではダメなのか？	決定木はピクセルごとの数値を単純に見るため、画像が少しズレただけで認識できなくなります。CNN のように「特徴」を見ることができないため、画像認識には不向きです。

§ 5. 確認ポイント

以下の観点で学生の理解度を確認します。

項目	確認ポイント
Part 2 考察	学習曲線を見て、「過学習」が起きていることをグラフの特徴（訓練と検証の乖離）から説明できているか
Part 3 考察	ドロップアウト導入後の学習曲線を比較し、過学習が抑制された（乖離が縮まった）ことを確認できているか
Part 4 最終評価	転移学習による精度の向上を数値で示し、その理由（事前学習モデルの知識利用）を説明できているか
実データテスト	自分の用意した画像での予測結果に対し、正解・不正解の理由（背景、画角、データとの違いなど）を考察できているか

人工知能（AI）技術応用
実習２　画像認識 AI で犬と猫を分類
Part１：データ準備と CNN 入門　教員用指導ガイド

§ 1. 本 Part の目的と到達目標

本 Part では、ディープラーニングを行うための環境（GPU）を整え、大量の画像データを AI が学習しやすい形に加工する「前処理」の技術を習得します。

到達目標

- ✅ Google Colab で **GPU ランタイム**を有効化し、TensorFlow/Keras 環境を準備できる
- ✅ 画像データ（カラー3 層）の構造を理解し、可視化して確認できる
- ✅ **正規化**と**データ拡張（Data Augmentation）**の実装目的を説明できる

§ 2. タスクごとの内容

タスク	指導・活動内容
導入・環境設定	実習 2 の概要説明。GPU 設定の確認（最重要）
データロード・確認	データのダウンロードと展開。画像を表示して「データの多様性」を確認
前処理・データ拡張	ImageDataGenerator の設定。データ拡張の効果を視覚的に確認
まとめ・考察	なぜ前処理が必要なのかを考察し、次回のモデル構築へ繋げる

§ 3. 詳細な指導ステップとポイント

Step 1: 環境設定（GPU の有効化）

ディープラーニングは計算量が膨大です。必ず GPU を使用します。

⚠️ **教員チェックポイント**生徒全員が「ランタイム」＞「ランタイムのタイプを変更」で「GPU」を選択しているか、目視で確認してください。ここが漏れると Part 2 以降で授業がストップします。

Step 2: データの確認（可視化）

ダウンロードした画像（犬と猫）を数枚表示させます。

💡 **生徒への問いかけ**「表示された画像を見てみよう。大きさはバラバラ？ 色はどう？ 背景は？」

→ AI に入力するためには、これらを統一（リサイズ）する必要があることに気づかせます。

Step 3: 画像の前処理（正規化とリサイズ）

ImageDataGenerator を使って、画像を AI 用データに変換します。

正規化（rescale=1./255）の理由：

画像のピクセル値（0〜255）をそのまま使うと数字が大きすぎて計算が不安定になります。0〜1 の範囲にギュッと縮めることで、AI がスムーズに学習できるようになります。

Step 4: データ拡張（Data Augmentation）の実装

画像を回転させたり、反転させたりして「水増し」します。

💡 **概念の解説（例え話）**「人間は、寝転がっている猫を見ても猫だと分かりますね？ でも AI は『立った猫』しか見たことがないと、『寝た猫』が分かりません。

だから、わざと画像を傾けたりひっくり返したりして、**『色々なポーズの猫』を擬似的に作って教える**のです。これがデータ拡張です。」

§ 4. よくあるトラブルと対処法

⚠️ **エラー1：GPU が見つからない / 学習が遅い**

症状：コード実行時に GPU 関連のエラーが出る、または処理が極端に遅い。

対処：ランタイム設定を再確認し、一度「セッションの管理」からセッションを終了して再接続させてください。

⚠️ **エラー2：画像が表示されない（FileNotFoundError）**

症状：データの読み込みや表示でエラーになる。

対処：データの解凍（unzip）が完了していない可能性があります。ファイル構成を確認し、必要なら解凍コマンドを再実行してください。

⚠️ **エラー3：メモリ不足（RAM オーバー）**

症状：セッションがクラッシュする。

対処：一度に大量の画像を表示しようとしすぎていないか確認。`batch_size`の設定（通常 32 程度）が守られているか確認してください。

§ 5. 確認ポイント

項目	確認ポイント
環境構築	GPU 環境で TensorFlow/Keras が正しく動作しているか
データ理解	訓練データと検証データの役割分担（勉強用とテスト用）を理解しているか
考察力	「なぜデータ拡張が必要なのか？」という問いに対し、過学習や汎化性能の観点から自分の言葉で説明できているか

人工知能（AI）技術応用
実習２　画像認識 AI で犬と猫を分類
Part２：CNN モデルの構築と学習　教員用指導ガイド

§ 1. 本 Part の目的と到達目標

本 Part では、ディープラーニングの核となる **CNN（畳み込みニューラルネットワーク）** をゼロから構築し、実際に学習させます。最大の学習目標は、学習曲線を通じて AI 開発の課題である**「過学習」**を発見することです。

到達目標

- ✅ CNN の基本構造（畳み込み層、プーリング層、全結合層）の役割を説明できる
- ✅ Keras を用いてモデルを構築・コンパイルし、学習を実行できる
- ✅ 学習曲線を可視化し、訓練データと検証データの乖離から「過学習」を見抜くことができる

§ 2. タスクごとの内容

タスク	指導・活動内容
導入・CNN 解説	CNN の仕組み（特徴抽出と圧縮）を直感的に解説
モデル構築	Keras で層を積み重ねるコーディング
モデル学習（最長）	.fit()実行。待ち時間に過学習の概念を説明
結果確認・考察	学習曲線のグラフ化と過学習の発見

§ 3. 詳細な指導ステップとポイント

Step 1: CNN の仕組みの解説（導入）

🧠 生徒への説明のヒント（例え話）

- **畳み込み層（Convolution）**：「虫眼鏡」で画像の一部を詳しく見て、縦線やカーブなどの「特徴」を見つける役割。
- **プーリング層（Pooling）**：画像を「モザイク」のようにぼかして縮小し、細かいズレを気にしなくする役割（情報の圧縮）。
- **全結合層（Dense）**：見つかった特徴（耳が尖ってる、ヒゲがある）を元に、「これは猫だ！」と

結論を出す脳の役割。

Step 2: モデルの構築（Keras）

Sequential モデルを使って層を積み上げます。特に「Flatten（平坦化）」の役割（2 次元の画像を 1 次元の数値列に変換し、全結合層に渡す）を忘れずに説明してください。

💡 **指導のアクション**

model.summary() を実行させ、パラメータ数（学習する変数の数）に注目させてください。「数百万個もの計算を自動で行うため、GPU が必要なんだ」と繋げると納得感が高まります。

Step 3: 学習の実行（待ち時間の活用）

学習（model.fit）には GPU 環境でも数分〜十数分かかります。この時間を有効活用しましょう。

💡 **待ち時間の指導アイデア**

- **過学習の予習:**「テスト勉強で過去問を丸暗記しただけの学生」の話をし、初見の問題（検証データ）が解けない状態＝過学習であることをイメージさせる。
- **ログの観察:**リアルタイムで更新される loss（間違い）が減り、accuracy（正解率）が上がっていく様子を実況させる。

Step 4: 学習曲線の分析（最重要）

学習完了後、グラフを表示させます。ここで必ず**「過学習」**が発生しているはずです。

📉 **過学習のサイン（グラフの見方）**

- **青線（訓練データ）：**右肩上がりで 100%に近づく。
- **オレンジ線（検証データ）：**途中から上がらなくなる、あるいは下がり始める。
- **結論:** 2 本の線の「乖離（ギャップ）」が開いていくのが過学習の証拠です。

§ 4. よくあるトラブルと対処法

⚠️ **エラー1: 学習が極端に遅い（1 エポックに数分かかる）**

原因: GPU が有効になっていない可能性が高いです。

対処: Colab の「ランタイム」＞「ランタイムのタイプを変更」で GPU を確認し、再接続してください。

⚠️ **エラー2: ResourceExhaustedError（メモリ不足）**

原因: 画像サイズやバッチサイズが大きすぎます。

対処: `batch_size`を 32 から 16 に減らすか、画像サイズを小さくしてください。

⚠️ **エラー3: グラフが表示されない**

原因: plt.show() の記述忘れや、変数名（history）の間違い。

対処: 前のセル（学習）が正常に完了し、history 変数に結果が格納されているか確認させてください。

§ 5. 確認ポイント

評価項目	確認ポイント
モデル構築	CNN の層（Conv2D, MaxPooling2D, Flatten, Dense）が正しい順序で記述されているか
学習の実行	GPU を利用してエラーなく学習を完走できたか
考察（重要）	学習曲線を見て、「訓練データと検証データの正解率に差が開いている（過学習）」ことを自分の言葉で指摘できているか

人工知能（AI）技術応用
実習２　画像認識 AI で犬と猫を分類
Part３：モデルの評価と改善　教員用指導ガイド

§ 1. 本 Part の目的と到達目標

本 Part では、Part ２で作成したモデルの「本当の実力」をテストデータで測定し、AI がどのような間違いをしたかを分析します。さらに、過学習対策として最もポピュラーな手法である**「ドロップアウト（Dropout）」**を実装し、モデルを改善します。

到達目標

- ✅ テストデータを用いて、モデルの客観的な正解率を評価できる
- ✅ 誤分類された画像（AI が間違えた画像）を確認し、その原因を考察できる
- ✅ ドロップアウト層を追加したモデルを再学習させ、過学習の抑制効果を学習曲線で確認できる

§ 2. タスクごとの内容

タスク	指導・活動内容
テストデータ評価	.evaluate()で最終スコアを確認。「模試」と「本番」の違いを説明
誤分類分析	AI が間違えた画像を表示し、人間が見て理由を考える
モデル改善（再学習）	ドロップアウト層を追加して再学習。待ち時間に概念解説
改善結果の確認	新しい学習曲線を比較し、過学習が減ったことを確認

§ 3. 詳細な指導ステップとポイント

Step 1: テストデータでの最終評価

Part ２の最後に確認した「検証データ（Validation）」の精度だけでなく、学習に全く使っていない「テストデータ（Test）」での精度を確認させます。

🎓 なぜテストデータが必要？（生徒への説明）

「検証データは、学習中に『今の調子どう？』と何度も確認するために使いました（模擬試験）。
AI は模擬試験の問題に少し慣れてしまっているかもしれません。
だから、一度も見たことがない『テストデータ（本番の入試）』で、本当の実力を測る必要がありま

す。」

Step 2: 誤分類画像の分析（AI の弱点探し）

AI が「犬を猫」あるいは「猫を犬」と間違えた画像を実際に表示させます。

💡 **考察を深める問いかけ**

- 「人間が見ても間違えそうな画像はある？」
- 「背景がごちゃごちゃしていると間違えやすい？」
- 「体の一部しか映っていない画像はどう？」

→ AI は「耳の形」や「毛並み」などの特徴を見ているため、それが隠れていると間違えやすいことに気づかせます。

Step 3: ドロップアウトの実装（過学習対策）

モデルに layers.Dropout(0.5) を追加して再学習させます。これは学習中にランダムに半分のニューロンを無効化する処理です。

🧠 **ドロップアウトのイメージ**

「優秀なエース社員（特定のニューロン）だけに頼るチームは、エースが休むと仕事ができません。ドロップアウトは、強制的にランダムにメンバーを休ませることで、**『誰がいても仕事ができる強いチーム』**を作るトレーニング方法です。

これにより、特定のデータ特徴に依存しすぎる（過学習）のを防ぎます。」

Step 4: 改善効果の確認

再学習後の学習曲線を確認します。Part 2 のグラフと比較させます。

💡 **成功の基準**

- 訓練データ（青線）と検証データ（オレンジ線）の**隙間（ギャップ）が狭まっていること**が成功の証です。
- 全体の正解率が劇的に上がらなくても、ギャップが縮まっていれば「汎用性が高まった（過学習が減った）」と言えます。

§ 4. よくあるトラブルと対処法

⚠️ **エラー1: 評価スコアが極端に低い（50%程度）**

原因: 学習がうまくいっていないモデルをロードしている、またはモデルの重みがリセットされている。

対処: Part 2 で保存したベストモデル（.h5 ファイル）を正しくロードできているか確認してください。

⚠️ **エラー2: 誤分類画像が表示されない**

原因: 予測結果の配列インデックスの指定ミス。

対処: コード内の predictions[i] と true_labels[i] の比較ロジックを確認してください。画像データジェネレータの shuffle=False 設定も重要です。

⚠️ **質問: ドロップアウトを入れたのに精度が下がった？**

回答: 学習しにくくする（負荷をかける）手法なので、訓練データの精度は下がることがあります。重要なのは「検証データの精度が安定しているか（下がっていないか）」です。

§ 5. 確認ポイント

評価項目	確認ポイント
評価の実施	テストデータを使って model.evaluate() を実行し、数値を記録できているか
分析力	誤分類画像の特徴（暗い、複数匹いる等）を具体的に挙げられているか
改善の理解	ドロップアウト導入前後の学習曲線を比較し、「過学習が抑制された（差が縮まった）」ことを指摘できているか

人工知能（AI）技術応用
実習２　画像認識 AI で犬と猫を分類
Part４：転移学習と実データテスト 教員用指導ガイド

§ 1. 本 Part の目的と到達目標

本 Part では、AI 開発の現場で最も強力な手法の一つである**「転移学習（Transfer Learning）」**を体験します。事前学習済みモデル（VGG16）を利用して短時間で高精度な AI を構築し、最後に学生自身が用意した画像で実用性をテストします。

到達目標

- ✅ 転移学習の仕組み（知識の再利用）とメリット（時短・高精度）を説明できる
- ✅ Keras で VGG16 モデルをロードし、ファインチューニング（層の追加と再学習）を実装できる
- ✅ 自分の画像で AI をテストし、AI の得意・不得意（データの偏りなど）を考察できる

§ 2. タスクごとの内容

タスク	指導・活動内容
導入・概念解説	「巨人の肩に乗る」比喩で転移学習を解説
転移学習の実装	VGG16 のロード、重みの固定、学習実行
実データテスト	自分の画像をアップロードし、予測させる
まとめ・レポート	Part 1〜4 の総括と考察レポート作成

§ 3. 詳細な指導ステップとポイント

Step 1: 転移学習の概念解説（巨人の肩に乗る）

🧠 **生徒への説明のヒント**

「Part 2 では、赤ちゃん（何もしらない AI）に犬と猫を教えました。
今回は、**『動物博士（VGG16）』**に手伝ってもらいます。
博士はすでに『毛並み』や『目』の特徴を知っています。だから、『こういう特徴のが犬だよ』と少し教えるだけで、すぐに天才的な識別ができるようになるのです。」

Step 2: VGG16 モデルの実装と学習

Keras から VGG16 をロードし、ベースモデルの重みを**「凍結（Freeze）」**する手順が重要です。

💡 **技術的なポイント**

- include_top=False: VGG16 の最後の分類層（1000 クラス用）は不要なのでカットします。
- trainable=False: 博士の知識（重み）を壊さないように、再学習を禁止します。
- **結果の比較:**Part 3（自作モデル）の精度（約 70-80%）と、今回の精度（90%以上）を黒板などで並べて比較させ、威力を実感させましょう。

Step 3: 自分の画像でテスト（実社会への応用）

学生にスマホなどで撮影した、またはネットで拾った「犬」「猫」の画像を Colab にアップロードさせます。

💡 **考察のチャンス：AI が間違えたら？**

「家の飼い猫を犬と間違えた！」→ チャンスです。

「なぜだろう？ VGG16 は『図鑑のような綺麗な写真』で勉強したから、生活感のある部屋の背景や、変なポーズに弱いのかもしれない」

このように、**学習データと実データのギャップ（ドメインシフト）**について考えさせると、非常に深い学びになります。

§ 4. よくあるトラブルと対処法

⚠️ **エラー1: VGG16 のダウンロードが終わらない**

原因: ネットワークの問題やファイルサイズ（約 500MB）の影響。

対処: Colab の再接続、または少し待つように指示。どうしてもダメな場合は、ダウンロード済みの重みファイルを共有フォルダからコピーさせる等の代替案を用意。

⚠️ **エラー2: 独自画像の予測エラー（Shape mismatch）**

原因: アップロードした画像の前処理（リサイズ、次元追加）忘れ。

対処: 画像を読み込んだ後、必ず（1, 150, 150, 3）の形に変形し、/ 255.0 で正規化しているかコードを確認させてください。

⚠️ **質問: なぜ学習がこんなに速いのですか？**

回答: ほとんどの層（VGG16 部分）は計算済みの答えを使うだけで、学習（更新）していないからです。最後の数層だけを計算すればいいので、非常に高速です。

§ 5. 確認ポイント

- **技術理解:** 転移学習において「何を固定し、何を学習させたか」を説明できるか
- **性能比較:** 自作モデルと転移学習モデルの精度差を数値で示せているか
- **考察力:** 実データテストの結果から、AI の特性（得意な画像、苦手な画像）を論理的に推測できているか

人工知能（AI）技術応用
実習 3：自然言語処理で感情分析 AI を作る
導入編 教員用指導ガイド

§ 1. 実習の目的と位置づけ

🎯 メインテーマ：テキストデータ（自然言語）の AI 活用

実習 1（数値）、実習 2（画像）に続き、第 3 の主要データ形式である「テキスト」を扱います。映画レビューの文章から感情（ポジティブ/ネガティブ）を判定する AI を構築し、自然言語処理（NLP）の基礎から最新技術（BERT）までを体験的に学びます。

学生の到達目標

- テキストを数値化する「**前処理（トークン化・パディング）**」の必要性を理解し、実装できる
- 文章の順序や文脈を理解する **RNN/LSTM モデル**を構築できる
- **Precision, Recall, F1 Score** などの指標を用いて、モデルの性能を多角的に評価できる
- 最新の **BERT モデル（転移学習）**を活用し、その強力さを体感する

これまでの実習との比較

項目	実習 1（数値）	実習 2（画像）	実習 3（テキスト）
データ形式	構造化データ	非構造化（ピクセル）	**非構造化（単語列）**
前処理	正規化	リサイズ、正規化	**トークン化、パディング**
モデル	決定木、k-NN 等	CNN	**RNN, LSTM, BERT**
難易度	入門	中級	**中級〜応用**

§ 2. 実習の全体構成（全 4 Part）

全実習時間は、講義・実習・考察を含め、**4 コマ（約 270 分）**を目安とします。

Part	テーマ	主な学習内容
1	**環境準備と**	• IMDb データセットの確認

Part	テーマ	主な学習内容
	テキスト前処理	• トークン化（単語→ID） • パディング（長さ統一）
2	**LSTM モデル構築と過学習の発見**	• 単語埋め込み（Embedding） • LSTM モデルの構築・学習 • 過学習の検出
3	**モデルの評価と改善**	• Precision/Recall/F1 Score 評価 • 誤分類分析 • ドロップアウトによる改善
4	**BERT 活用と実データテスト**	• Transformer/BERT の概念 • 転移学習（ファインチューニング） • 実データでの最終テスト

§ 3. 指導上の工夫と重要ポイント

① テキスト前処理の「比喩」を使う

テキストの数値化は抽象的で難解です。以下の比喩を使って説明すると効果的です。

💡 **指導のヒント**

- **トークン化：**「辞書を作って、単語に背番号を振る作業」
- **パディング：**「原稿用紙のマス目を埋める作業（足りない部分は空白=0 で埋める）」
- **埋め込み層：**「単語の意味マップを作る作業（似た単語は近くに配置）」

② GPU 環境の徹底確認

⚠️ **最重要：GPU の設定**

LSTM や BERT は計算量が非常に多く、CPU では学習が終わりません。授業開始時に必ず、Google Colab の「ランタイムのタイプ」が GPU になっているか確認させてください。

③ BERT の「衝撃」を演出する

Part 4 の BERT は実習のハイライトです。Part 2 の自作 LSTM モデルとの性能差（精度、学習効率）を比較させ、「巨人の肩に乗る（事前学習モデルを使う）」ことの威力を実感させましょう。

§ 4. よくあるトラブルと対処法

症状・質問	原因と対処法
英語のデータセットしかないの？	今回は世界標準の IMDb（英語）を使います。日本語は単語の区切り（分かち書き）が難しく、専用ツール（MeCab 等）が必要になるため、入門としては英語データの方が構造を理解しやすい旨を説明してください。
学習が進まない / 遅い	GPU が有効になっていない可能性が高いです。設定を確認し、再接続してください。また、BERT の初回ダウンロードには時間がかかるため、少し待つよう指示してください。
「トークン化」後のデータがただの数字に見える	正常です。AI は言葉の意味ではなく、数値のパターンとして学習することを強調してください。デコード（数字→単語）して元に戻せることもデモで見せると納得感が高まります。

§ 5. 確認ポイント

以下の観点で学生の理解度を確認します。

項目	確認ポイント
ワークブック完成	Part 1〜4 の全コードを実行し、エラーなく完了しているか。
考察問題	「なぜパディングが必要か？」「なぜ BERT は高精度なのか？」といった問いに対し、技術的な根拠を持って回答できているか。
発展課題	自作のレビュー文でテストを行ったり、ハイパーパラメータの変更を試したりしているか。

人工知能（AI）技術応用
実習３　自然言語処理で感情分析 AI を作る
Part１：環境準備とデータ確認　教員用指導ガイド

§ 1. 本 Part の目的と到達目標

本 Part では、自然言語処理（NLP）の第一歩として、コンピュータが「言葉」を理解できるようにするための**「前処理（数値化）」**の技術を習得します。

到達目標

- ✅ 映画レビューデータセット（IMDb）をロードし、テキストデータとラベル（感情）の関係を確認する
- ✅ NLP 特有の 3 つの前処理ステップ（トークン化、ID 変換、パディング）を実装し、その目的を説明できる
- ✅ テキストデータの特殊性（可変長、順序重要）を理解する

§ 2. タスクごとの内容

タスク	指導・活動内容
導入・データロード	NLP の概要説明。IMDb データの確認と、テキストが数値リストで表されることの確認
トークン化・ID 変換	「単語」を「ID」に変換する仕組み（辞書）の解説とデモ
パディング	文の長さを揃える処理の実装。なぜ長さ統一が必要かを解説
まとめ・考察	「前処理」の意義を確認し、次回のモデル構築へ繋げる

§ 3. 詳細な指導ステップとポイント

Step 1: データの確認（言葉は数値？）

データセットをロードすると、レビューはいきなり数値のリストとして表示されます。

💡 **生徒への問いかけ**「あれ？映画のレビューなのに、数字しか書いてないね。なぜだろう？」

コンピュータは言葉の意味を直接理解できないため、あらかじめ単語に番号（ID）が振られているこ

とを気づかせます。

デモの推奨： ID リストをデコードして、元の英語の文章（例: "This movie was bad"）に戻して見せることで、数値と言葉の対応関係を直感的に理解させましょう。

Step 2: トークン化と ID 変換の理解

文章を単語（トークン）に区切り、ID に変換するプロセスを解説します。

💡 **比喩による説明**「学校の出席番号のようなものです。『田中さん』＝『1 番』、『佐藤さん』＝『2 番』と決めておけば、名前を言わなくても番号だけで誰か分かりますね。AI も同じように単語を管理しています。」

Step 3: パディング（長さの統一）

短い文には「0」を足し、長い文はカットして、長さを揃えます。

💡 **なぜこれが必要？**「AI（特に次の Part で使う LSTM）は、決まった長さのデータしか受け取れません。
短い作文用紙には空白（0）を入れて埋め、長すぎる作文は枠に収まるように切る必要があるのです。」

§ 4. よくあるトラブルと対処法

⚠️ **エラー1: データのダウンロード失敗**

症状: IMDb データのロード時にネットワークエラーが発生する。

対処: インターネット接続を確認し、セルを再実行してください。Keras のデータセットは初回のみダウンロードに時間がかかります。

⚠️ **エラー2: メモリエラー / 処理が重い**

症状: パディング処理などで Colab がフリーズする。

対処: 使用する単語数（`num_words`）を 10,000 から 5,000 に減らすか、最大長（`maxlen`）を短くしてみてください。

⚠️ **質問: なぜ英語のデータなのですか？**

回答: 英語は単語がスペースで区切られているため、日本語のような複雑な処理（分かち書き）が不要で、初心者が構造を理解しやすいからです。

§ 5. 確認ポイント

項目	確認ポイント
前処理の理解	トークン化、ID 変換、パディングの 3 ステップを正しい順序で実行できているか
データの確認	パディング後のデータ形状（例：25,000 行 × 256 列）を確認し、その意味を理解しているか

考察力	「画像データ（実習 2)」と「テキストデータ（実習 3)」の前処理の違いについて、自分の言葉で比較・考察できているか

人工知能（AI）技術応用
実習３　自然言語処理で感情分析 AI を作る
Part２：RNN/LSTM モデルの構築　教員用指導ガイド

§ 1. 本 Part の目的と到達目標

本 Part では、Part 1 で準備したデータを用いて、文章の「意味」と「順序」を理解できるディープラーニングモデル（LSTM）を構築し、学習させます。

到達目標

- ✅ **単語埋め込み（Embedding）**の概念（単語を意味ベクトルにする）を理解し、実装する
- ✅ 文章の順序を学習できる **LSTM（Long Short-Term Memory）**層を構築する
- ✅ モデルの学習（.fit()）を実行し、学習曲線から**過学習**を発見する

§ 2. タスクごとの内容

タスク	指導・活動内容
Embedding 解説	単語を数値ベクトルに変換する仕組みを直感的に解説
モデル構築	Keras で LSTM を含むモデルをコーディング
モデル学習	学習実行（待ち時間 10-15 分）。待ち時間の活用（Q&A）
結果分析	学習曲線の確認と「過学習」の議論

§ 3. 詳細な指導ステップとポイント

Step 1: 単語埋め込み（Embedding）の解説

Part 1 の「単語 ID」を「意味のあるベクトル」に変換する層です。

💡 **生徒への問いかけ**「『犬』と『猫』は似ているけど、『犬』と『車』は全然違うよね？でも ID（数字）だと、単なる番号の違いしかありません。Embedding 層は、単語同士の**『意味の距離』**を学習して、似た単語を近くに配置する地図を作るようなものです。」

Step 2: LSTM モデルの構築

Embedding → LSTM → Dense というシンプルな構造を作ります。

💡 **LSTM の役割（比喩）**「LSTM は**『記憶力のある読書家』**です。文章を最初から読んでいき、『not』のような重要な単語が出てきたら、『おっと、否定形が来たぞ』と記憶しておき、後の単語の意味を正

しく解釈します。」

Step 3: 学習の実行（待ち時間の活用）

LSTM の学習は計算時間がかかります（GPU 必須）。

⚠️ **学習時間の目安** GPU を使用しても、1 エポックあたり数分かかる場合があります。この時間を使い、次のタスクの説明や、実習 1・2 の復習クイズを行うなどして、学生の集中力を維持してください。

Step 4: 過学習の発見

学習曲線をグラフ化し、訓練データ（青線）と検証データ（赤線）の乖離を確認します。

観察ポイント： 検証データの正解率（val_accuracy）が途中で上がらなくなったり、逆に下がったりする現象を見つけさせます。「勉強しすぎて応用が効かなくなった状態」として説明します。

§ 4. よくあるトラブルと対処法

⚠️ **エラー1: 学習が極端に遅い（CPU 実行）**

症状: 1 エポックに数十分かかる。

対処: ランタイムが GPU になっていない可能性が高いです。設定を変更し、最初から実行し直すのが最善です。

⚠️ **エラー2: メモリエラー（OOM）**

症状: セッションがクラッシュする。

対処: バッチサイズ（`batch_size`）を 128 から 64 や 32 に減らす、または使用する単語数（`num_words`）を減らしてみてください。

⚠️ **質問: LSTM と RNN の違いは？**

回答: RNN は「短期記憶」しか持てず、長い文章だと最初のほうを忘れてしまいます。LSTM は「長期記憶」を持つことができる改良版で、長いレビューでも文脈を保つことができます。

§ 5. 確認ポイント

項目	確認ポイント
モデル構築	Embedding 層と LSTM 層を含むモデルを正しく定義し、コンパイルできているか
学習の実行	GPU を利用して学習を完了させ、その結果（正解率など）を記録しているか
考察力	学習曲線を見て「過学習」が発生しているかどうかを判断し、その理由を説明できているか

人工知能（AI）技術応用
実習３　自然言語処理で感情分析 AI を作る
Part３：モデルの評価と可視化　教員用指導ガイド

§ 1. 本 Part の目的と到達目標

本 Part では、Part 2 で作成した LSTM モデルの「本当の実力」をテストデータで測定し、正解率だけでなく Precision（適合率）や Recall（再現率）といった多角的な指標で評価・分析します。

到達目標

- ✅ テストデータを用いて、モデルの客観的な正解率を評価できる
- ✅ Precision, Recall, F1 Score の意味を理解し、モデルの性能を説明できる
- ✅ 誤分類（ポジティブをネガティブと判定など）の理由を文章の内容から考察できる

§ 2. タスクごとの内容

タスク	指導・活動内容
テストデータ評価	.evaluate() で最終スコアを確認。過学習の影響を数値で再確認
詳細評価（指標）	Precision, Recall の概念解説と算出。分類レポート（classification_report）の読み方
誤分類分析	AI が間違えたレビューを読み、「皮肉」や「二重否定」などの難しさを発見する

§ 3. 詳細な指導ステップとポイント

Step 1: テストデータでの評価

学習に使わなかった「テストデータ」で評価します。

💡 **生徒への問いかけ**「訓練データの正解率は高かったのに、テストデータの正解率が低いのはなぜでしょう？」

→ 過学習（勉強した問題しか解けない状態）であることを思い出させます。

Step 2: 評価指標の解説（Precision/Recall）

正解率（Accuracy）だけでは見えない弱点を探します。

指標の意味（比喩）：

- **Precision（適合率）：**「AI が『ポジティブだ！』と言ったもののうち、本当に合っていた確率」（オオカミ少年の嘘の少なさ）

- **Recall（再現率）：**「本当のポジティブレビューのうち、AI が見つけ出せた確率」（見落としの少なさ）

Step 3: 誤分類の分析（AI の弱点探し）

AI が間違えた文章を実際に読んでみます。

💡 **分析のヒント**「『悪い映画だとは思わない（not bad）』のような表現や、『期待外れではなかったが最高でもない』といった曖昧な表現を AI はどう判定しているでしょうか？」

→ 文脈や否定語の理解の難しさに気づかせます。

§ 4. よくあるトラブルと対処法

⚠️ **エラー1: 評価指標の計算エラー**

症状: classification_report 実行時にエラーが出る。

対処: sklearn.metrics のインポート漏れ、または予測結果を 0/1 の整数に変換（(pred > 0.5).astype(int)）していない可能性があります。

⚠️ **エラー2: 誤分類データが見つからない**

症状: 間違えたデータのインデックス抽出がうまくいかない。

対処: テストデータのラベル（`y_test`）と予測結果の形状（shape）が一致しているか確認させてください。

⚠️ **質問: Precision と Recall、どっちが大事？**

回答: 目的によります。スパムメール検知なら「必要なメールを捨てない（Recall 重視）」、医療診断なら「病気を見逃さない（Recall 重視）」など、状況によることを説明してください。

§ 5. 確認ポイント

項目	確認ポイント
客観的評価	テストデータでの正解率を記録し、訓練データとの差（過学習の度合い）を認識できているか
指標の理解	F1 Score が Precision と Recall の調和平均であることを理解し、モデルのバランスを評価できているか
定性分析	誤分類された具体的なレビュー文を挙げ、なぜ AI が間違えたのか仮説（皮肉、未知の単語など）を立てられているか

人工知能（AI）技術応用
実習３　自然言語処理で感情分析 AI を作る
Part４：事前学習モデル（BERT）の活用　教員用指導ガイド

§ 1. 本 Part の目的と到達目標

本 Part では、現代の自然言語処理のスタンダードである **BERT（トランスフォーマーモデル）**を活用し、転移学習によって劇的な性能向上を体験します。

到達目標

- ✅ **BERT** の概念（文脈の双方向理解）と、LSTM との違いを理解する
- ✅ **Hugging Face Transformers** ライブラリを使用し、事前学習済みモデルをロードする
- ✅ **転移学習（ファインチューニング）** を実装し、短時間の学習で高精度を達成する

§ 2. タスクごとの内容

タスク	指導・活動内容
BERT 解説	BERT がなぜ「賢い」のか、Attention（注意機構）の概念を直感的に解説
環境準備・ロード	ライブラリのインストールとモデルのロード（待ち時間あり）
ファインチューニング	BERT を用いた再学習。LSTM との学習速度・精度の違いを体感
実データテスト	自作のレビュー文で判定テストを行い、考察する

§ 3. 詳細な指導ステップとポイント

Step 1: BERT とトランスフォーマーの解説

LSTM は「前から順に」読みますが、BERT は「文章全体を一度に」見ることができます。

💡 **生徒への比喩**「LSTM は、本を 1 文字ずつ指で追いながら読む人です。
BERT は、ページ全体をパッと見て、単語同士の関係（文脈）を一瞬で把握できる速読の達人です。
しかも、BERT はインターネット上の大量の文章を読んで勉強済み（事前学習）のエリートです。」

Step 2: BERT モデルのロードと再学習

Hugging Face の `TFBertForSequenceClassification` を使用します。

ファインチューニングの手順：

1. 賢い BERT モデルをロードする。
2. 今回のタスク（映画レビューのポジネガ判定）に合わせて、少しだけ追加で勉強させる（微調整＝Fine-tuning）。
3. ゼロから学習するより圧倒的に早く、賢くなることを確認させる。

Step 3: LSTM vs BERT 性能比較

Part 3 の LSTM モデルの結果と比較します。

💡 **比較の視点**「同じデータを使っているのに、なぜここまで精度が違うのでしょうか？
それは、BERT が『言葉の裏にあるニュアンス』まで理解しているからです。
例えば、『面白い映画だとは思わなかった』という二重否定も、BERT なら正しくネガティブと判定できる可能性が高いです。」

§ 4. よくあるトラブルと対処法

⚠️ **エラー1: ライブラリが見つからない**

症状: `ModuleNotFoundError: No module named 'transformers'`

対処: Colab のセルで `!pip install transformers` を実行し忘れています。必ず最初に実行させてください。

⚠️ **エラー2: メモリエラー（OOM）**

症状: BERT は巨大なモデルなので、GPU メモリが不足することがある。

対処: バッチサイズ（`batch_size`）を 16 や 8 まで下げてください。学習速度は落ちますが動作します。

⚠️ **質問: なぜ学習データが英語なのですか？**

回答: 今回使用している BERT モデル（`bert-base-uncased`）が英語専用だからです。日本語を扱うには日本語用の BERT モデルが必要ですが、基本的な使い方は同じです。

§ 5. 確認ポイント

項目	確認ポイント
技術理解	転移学習（ファインチューニング）のプロセスを理解し、コードを実行できているか
性能比較	LSTM モデルと BERT モデルの精度（Accuracy/F1 Score）を数値で比較できているか
考察力	BERT が高性能である理由（事前学習、Attention 機構など）を、自分の言葉で説明できているか

ＩｏＴ技術応用

IoT 技術応用
実習 1　デバイス制御
導入編　教員用指導ガイド

§ 1. IoT 技術応用教材の位置づけ

IoT 技術基礎教材との関係

この教材は、『IoT 技術基礎』の**応用教材**として設計されています。

この教材の役割

IoT 技術基礎教材で学んだ理論を、実際のハードウェアとプログラミングで**体験・確認**します。

IoT 技術基礎	IoT 技術応用
理論・知識の習得	実践・体験による確認
「知る」「わかる」	「できる」「使える」
体系的・網羅的	重点的・実践的
個人学習	協働学習

教育的意義

理論と実践

この応用教材の最大の目的は、**理論と実践をつなぐことで、深い理解を促進すること**です。

学習の流れ

1. **基礎教材で理論を学ぶ**
    - IoT システムの構成要素
    - センサーやアクチュエータの仕組み
    - データ処理や通信の基礎
2. **実習で実践する**
    - 実際に回路を組む
    - プログラムを書いて動かす
    - 「なるほど、こう動くのか」と体感

実習の設計思想

段階的な学習

実習 1 の 4 回の実習は、以下の方針で設計されています：

- **Part1**：出力（LED）– 最も基本的な制御

- **Part2**：入力（センサー）－ データ取得の基礎
- **Part3**：統合（条件分岐）－ 入出力の組み合わせ
- **Part4**：完成（記録・表示）－ 実用システムの構築

各回で新しい概念を**一つずつ**追加することで、無理なく理解を積み上げます。

実用性の重視

最終的に作成するシステムは、「教材用の玩具」ではなく、**実際の現場で使える実用システム**です。

- 温室の環境管理
- 倉庫の温度記録
- 研究室のデータ収集

「学んだことが本当に役立つ」という実感が、学習意欲を高めます。

達成感の演出

各回で「動いた！」「できた！」という**成功体験**を積み重ねることを重視しています。

§ 2. IoT 技術応用 実習 1

実習 1 の 4 回の実習マップ

回	テーマ	新規概念	対応基礎教材
Part1	LED 制御	・デジタル出力 ・プログラム基礎	第 6 章、第 7 章
Part2	センサー	・センサー入力 ・ライブラリ ・シリアル通信	第 1 章、第 4 章、第 6 章
Part3	条件制御	・条件分岐 ・システム統合	第 1 章、第 3 章、第 7 章
Part4	記録・表示	・SD カード ・LCD ・SPI/I2C 通信	第 4 章、第 5 章、第 6 章

学習目標の体系

知識面（オンライン教材の確認）

Part	確認する理論知識
Part1	• IoT デバイスの基本構成 • アクチュエータの役割

	• プログラムの基本構造
Part2	• センサーの種類と原理 • データ収集の基礎 • 通信の基本
Part3	• エッジでのデータ処理 • リアルタイム制御 • システム統合
Part4	• データ保存方式 • 通信プロトコル（SPI、I2C） • データ可視化

技能面（実践スキル）

- 電子回路の配線
- Arduino プログラミング
- デバッグとトラブルシューティング
- システム統合

態度面（学習姿勢）

- 試行錯誤しながら学ぶ姿勢
- 問題を自分で解決する力
- 協働して学ぶ態度
- 理論と実践を結びつける習慣

段階的な難易度設計

認知負荷の管理

各回で学生が処理すべき情報量を適切に管理しています：

Part	新規要素数	配線の複雑さ	プログラムの長さ
Part1	3 個	簡単（3 本）	短い（20 行）
Part2	3 個	簡単（3 本追加）	中（30 行）
Part3	1 個	変化なし	中（40 行）
Part4	4 個	複雑（14 本）	長い（70 行）

ポイント：Part3 で一度負荷を下げることで、Part4 の複雑さに対応できる準備をします。

§ 3. 指導の基本方針

学生の主体性を引き出す

「教え込む」より「引き出す」

一方的に教えるのではなく、学生自身に考えさせ、気づかせることを重視します。

効果的な質問例

理解を確認する質問：

- 「なぜこの配線が必要だと思いますか？」
- 「このプログラムは、どんな順番で実行されますか？」
- 「オンライン教材のどこで学んだ内容ですか？」

思考を促す質問：

- 「もし抵抗を入れなかったら、どうなると思いますか？」
- 「温度が 30 度以上のとき、LED をどう動かせばいいでしょう？」
- 「このシステムを、どんな場面で使えそうですか？」

応用を促す質問：

- 「湿度でも同じように制御するには、どうすればいいでしょう？」
- 「複数のセンサーを使うには、何を追加すればいいですか？」

失敗を学びに変える

エラーは学習のチャンス

エラーや失敗を否定的に扱わず、**学習の機会**として活用します。

エラー対応の基本姿勢

1. 落ち着かせる

「エラーは誰にでも起こります。一緒に見てみましょう」

2. 自分で考えさせる

「エラーメッセージには何と書いてありますか？」

「どこを確認すればいいと思いますか？」

3. ヒントを与える

「配線を確認してみましょう」

「プログラムの〇行目を見てください」

4. 解決したら価値づける

「自分で見つけられましたね！素晴らしい」

「このエラーの対処法、ワークブックにメモしておくといいですよ」

典型的なエラーの教育的活用

エラー	学習の機会
LED の向き間違い	極性の概念を理解
セミコロン忘れ	プログラム文法の重要性

配線ミス	回路図の読み方の習得
論理エラー	デバッグ技術の習得

§ 4. オリエンテーションの実施

実施タイミング

オリエンテーションは、以下のいずれかのタイミングで実施します：

パターン 1：独立したオリエンテーション（推奨）

- **時期**：Part1 の 1 週間前
- **時間**：30-45 分
- **内容**：実習ガイドの説明、Arduino IDE のインストール確認
- **利点**：Part1 で実習にすぐ取りかかれる

パターン 2：Part1 の導入として実施

- **時期**：Part1 の最初
- **時間**：15-20 分
- **内容**：実習ガイドの要点説明
- **注意**：Part1 の実習時間が短くなる

オリエンテーションの内容と時間配分

独立オリエンテーション（45 分）の場合

項目	時間	内容
1. 実習の位置づけ	5 分	オンライン教材との関係、実習の目的
2. 全体像の説明	10 分	4 回の流れ、完成システムのデモ
3. 実習の進め方	10 分	各回の流れ、評価方法、注意事項
4. Arduino IDE 確認	10 分	インストール確認、起動テスト
5. 質疑応答	10 分	質問受付、次回の予告

Part1 導入オリエンテーション（20 分）の場合

項目	時間	内容
1. 実習の位置づけ	3 分	オンライン教材との関係（簡潔に）

2. 4回の流れ	5分	全体像を図で示す
3. 実習の進め方	7分	ワークブックの使い方、評価方法
4. 今日の予告	5分	Part1の目標、デモ

オリエンテーション実施のポイント

1. 実習の位置づけ（最重要）

話す内容：

- 「この実習は、オンライン教材で学んだ理論を実践で確認するためのものです」
- 「オンライン教材が『知る』なら、実習は『できる』です」
- 「両方を行き来することで、深い理解につながります」

視覚的に示す：

プロジェクターで以下の図を表示：

オンライン教材（理論）

⬇ 実践で確認

実習（体験）

⬇ 理論で理解

オンライン教材（復習）

⬇

深い理解と応用力

2. 完成システムのデモ

準備：

- Part4完成品を用意
- LCDをカメラで拡大表示
- シリアルモニタも表示
- SDカードの中身も見せる

デモの流れ：

1. 「これが、4回の実習で作るシステムです」
2. LCDに温度・湿度が表示されている様子を見せる
3. 「2秒ごとに自動で測定・表示されます」
4. 「温度が高いと、LEDの点滅が速くなります」
5. SDカードを取り出して、パソコンで開く
6. 「このように、データが自動で記録されています」
7. 「実際の農業や倉庫管理で使われているシステムと同じです」

学生の反応を引き出す：

- 「すごいでしょう？これを4回で作ります」
- 「難しそう？大丈夫、段階的に進めます」

- 「これができたら、自分でいろいろ作れるようになりますよ」

3. 実習の進め方

ワークブックの説明：

- 実物を見せながら説明
- 「各回にワークブックを配ります」
- 「手順が詳しく書いてあるので、見ながら進められます」
- 「実験結果や気づきを書く欄があります」
- 「考察問題も含まれています」

評価方法の説明：

- 「技能、知識、思考、態度の 4 つの観点で評価します」
- 「完璧にできることより、試行錯誤しながら学ぶ姿勢が大切です」
- 「エラーが出ても大丈夫。それも学習です」
- 「周りの人と協力することも評価対象です」

4. Arduino IDE の確認

実施手順：

1. 「それでは、Arduino IDE が正しくインストールされているか確認しましょう」
2. 「パソコンを起動してください」
3. 「Arduino IDE を起動してください」
4. 巡回して、起動できない学生をサポート
5. 「起動できましたか？では、簡単なプログラムを試してみましょう」
6. 「ファイル」→「スケッチ例」→「01.Basics」→「Blink」を開く
7. 「検証ボタン（チェックマーク）をクリックしてください」
8. 「エラーが出なければ OK です」

トラブル対応：

- インストールされていない → その場でインストール支援
- 起動しない → PC の設定確認、再インストール
- 検証エラー → Arduino IDE のバージョン確認

配布資料

オリエンテーション時に配布

- 実習ガイド（学生用）- 全員に配布
- 実習スケジュール - 掲示または配布
- Arduino IDE インストールガイド（未インストール者用）

オリエンテーション後のフォローアップ

- Arduino IDE インストール状況の確認（メール等）
- 質問受付の案内（メールアドレス、オフィスアワー）
- Part1 の予習範囲の案内（オンライン教材の該当章）

§ 5. 事前準備

全体の準備

部品の調達

- 必要な部品リストの作成
- 学生数×1.2 セット分を発注（予備込み）
- 納品確認と動作テスト
- 部品の仕分けと梱包

必要部品リスト（学生 1 人につき）

部品名	数量	備考
Arduino Uno	1	互換品でも可
ブレッドボード	1	400 穴以上（Part 1 は 5 列でも可）
LED（赤）	1	5mm、予備 2 個
抵抗 220Ω	1	予備 2 個
DHT11 センサーモジュール	1	予備 1 個 （部品単体の場合、プルアップ抵抗を用意）
SD カードモジュール	1	SPI 接続
microSD カード	1	32GB 以下、FAT32 フォーマット
LCD ディスプレイ	1	16×2、I2C 接続　（信号線注意）
ジャンパーワイヤー	20 本	オス-オス
USB-A to B ケーブル	1	学生持参も可

予備部品（教員用）

- 各部品の予備（学生数の 20%分）
- Arduino Uno 予備機（5-10 台）
- SD カードリーダー（複数個）
- テスター（配線確認用）

教材準備

- 実習ガイドの印刷（学生数分）

- Part1-4 のワークブック印刷（学生数分×4 回）

教室環境の準備

設備の確認

- 電源コンセントの数と配置
- プロジェクター/大型ディスプレイ
- 書画カメラ（配線を拡大表示用）
- ホワイトボード/黒板
- Wi-Fi 環境（Arduino IDE 更新用）

座席配置

推奨配置：ペアまたは 4 人グループ

- 相互に教え合える距離
- 教員が巡回しやすい配置
- 全員がプロジェクターを見られる

工具・消耗品

- プラスドライバー（小型）
- ピンセット
- クリーニングクロス
- 結束バンド
- ラベルシール

デモ用システムの準備

各回のデモ機

各回の完成形を事前に作成しておきます。

- Part1 完成形（LED 点滅）
- Part2 完成形（温湿度測定）
- Part3 完成形（条件制御）
- Part4 完成形（記録・表示）

デモ機の活用

- **導入時：**完成形を見せてモチベーション向上
- **説明時：**配線を拡大表示
- **トラブル対応時：**正しい動作を確認
- **巡回時：**持ち歩いて個別に見せる

デモ用データの準備

- SD カードに記録されたサンプルデータ
- Excel でグラフ化した例
- 応用例の写真や動画

サンプルプログラムの準備

配布用ファイル

プログラム入力に時間がかかる学生用に、サンプルプログラムを準備します。

- Part1-4 の完成プログラム
- 中間段階のプログラム（デバッグ用）
- 発展課題のサンプル

配布方法

- USB メモリで配布
- 学内ネットワークの共有フォルダ
- メール添付
- クラウドストレージ（Google Drive 等）

注意：学習効果を考慮し、**最初からファイルを配布せず**、時間が足りない学生や困っている学生にのみ提供することを推奨します。

§ 6. 各回の実習前準備

数日前の準備

- 該当回のワークブック印刷
- 該当回のデモ機動作確認
- 必要部品の在庫確認
- 予備部品の準備
- 教員用指導マニュアルの確認
- ナレーションスクリプトの確認

前日の準備

- 部品の動作確認（特にセンサー、モジュール）
- デモ機の最終確認
- プロジェクター接続確認
- 書画カメラ動作確認
- ワークブックの配布準備
- 板書内容の準備

Part4 の特別準備

- 全 SD カードを FAT32 でフォーマット
- LCD の I2C アドレス確認（0x27 or 0x3F or 0x50 …）
- SD カードリーダーの準備

部品配布の工夫

- **Part1**：Arduino、ブレッドボード、LED、抵抗、ワイヤー
- **Part2**：DHT11 センサー、ワイヤー追加（Part1 はそのまま）
- **Part3**：新規部品なし（Part2 の回路そのまま）
- **Part4**：SD カード、LCD、ワイヤー追加（Part3 はそのまま）

💡 **ヒント**：部品を透明袋に入れて配布すると、紛失防止と管理がしやすい

§ 7. 指導上の注意点

安全管理

基本的な注意事項

- 配線時は必ずUSBケーブルを抜くよう指導
- 部品を無理に押し込まないよう注意
- 発熱している部品に触らないよう注意
- 静電気対策（冬季は特に注意）

トラブル発生時の対応

- **発煙・異臭**：即座に電源を切る
- **部品の発熱**：配線ミスを疑う
- **破損**：予備部品と交換

§ 8. トラブル対応

よくあるトラブルと対処法

Arduino IDEのトラブル

症状：プログラムが書き込めない

原因と対処：

- USBケーブル接続不良 → ケーブルを差し直す
- ポート設定が間違っている → 「ツール」→「シリアルポート」で確認
- ボード設定が間違っている → 「Arduino Uno」を選択
- ドライバ未インストール → ドライバをインストール

症状：コンパイルエラーが出る

原因と対処：

- セミコロン忘れ → エラー行を確認して追加
- 括弧の対応ミス → 括弧の数を確認
- ライブラリ未インストール → ライブラリをインストール
- スペルミス → エラーメッセージを確認

配線トラブル

LED関連

症状：LEDが点灯しない

- 極性が逆 → LEDの向きを確認（長い足がアノード）
- 抵抗なし → 220Ω抵抗を挿入
- 配線ミス → ピン番号を確認
- LED破損 → 予備と交換

センサー関連

症状：温湿度が取得できない

- 配線ミス → VCC、GND、DATAピンを確認
- ピン番号間違い → プログラムと配線を照合

- センサー不良 → 予備と交換
- ライブラリ未インストール → DHT ライブラリをインストール

SD カード関連

症状：SD カード初期化失敗

- SD カード未挿入 → カチッと音がするまで押し込む
- 配線ミス（特に CS） → D10 に接続されているか確認
- SD カードフォーマット → FAT32 で再フォーマット
- 容量が大きすぎる → 32GB 以下のカードを使用

LCD 関連

症状：LCD に何も表示されない

- 配線ミス → SDA が A4、SCL が A5 か確認（D4/D5 ではない）
- 電源不足 → VCC が 5V に接続されているか確認
- I2C アドレス違い → 0x27 を 0x3F に変更して試す
- コントラスト調整 → 裏の可変抵抗を回す
  （※秋月電子通商で入手した場合は、利用方法をホームページで要確認）

部品破損への対応

破損の判断

以下の症状があれば、部品破損を疑います：

- 配線が正しいのに動作しない
- 他の学生の部品と交換すると動く
- 異臭や発熱がある
- 物理的な破損が見える

交換手順

1. 電源を切る
2. 破損部品を取り外す
3. 予備部品と交換
4. 配線を確認
5. 動作確認

破損記録

部品破損は記録しておき、次回の調達に反映します。

- 部品名
- 破損日時
- 破損原因（わかれば）
- 対処方法

学生の心理的サポート

失敗への恐れがある学生

「間違えたらどうしよう」と不安で手が動かない学生への対応：

- 「失敗しても大丈夫。予備があります」と安心させる
- 「エラーは学習のチャンス」と価値づける

- 小さな成功体験を積ませる（まずLEDだけ点灯など）
- 他の学生の成功例を見せる

挫折しそうな学生

「自分にはできない」と諦めかけている学生への対応：

- これまでできたことを確認する
- 「ここまではできています」と認める
- 一緒に一つずつ確認する
- 周りの学生のサポートを依頼
- 最終的には必ず動かして成功体験を

周りと比較して焦る学生

「自分だけ遅れている」と焦っている学生への対応：

- 「自分のペースで大丈夫」と伝える
- 「速さより理解が大切」と価値づける
- その学生の成長を認める
- 時間延長の配慮

§ 9. 実習後のフォロー

ワークブックの返却と復習

ワークブックの確認項目

- 全ての記入欄が埋まっているか
- 実験結果は正確に記録されているか
- 考察は適切か
- 自己評価は妥当か
- 質問欄に記入があれば個別対応

フィードバックのコメント

各学生のワークブックに、コメントを書いて返却します。

良いコメント例：

- 「オンライン教材の内容と結びつけて考察できていますね」
- 「エラーに対して粘り強く取り組んでいました」
- 「独自の気づきが素晴らしいです」
- 「次回は○○にも注目してみてください」

返却のタイミング

- 可能であれば次回実習の前まで
- コメントを読んで復習できる時間を確保

§ 10. 参考情報

部品の調達情報

推奨サプライヤー

- **国内：**秋月電子通商、千石電商、マルツ

- **海外：**AliExpress、Amazon

購入のポイント

- 学生数の 1.2 倍前後を購入（予備込み）
- 早めに発注（納期に余裕を持つ）
- 動作確認を必ず実施
- 不良品は返品・交換

関連資料

Arduino 関連

- Arduino 公式サイト：https://www.arduino.cc/
- Arduino 日本語リファレンス：http://www.musashinodenpa.com/arduino/ref/
- Arduino Project Hub：https://create.arduino.cc/projecthub

教育関連

- 産業技術総合研究所 IoT 教育資料
- 経済産業省 IoT 推進ラボ
- 総務省 IoT 教育推進コンソーシアム

コミュニティ

- Arduino Forum：https://forum.arduino.cc/
- Stack Overflow（Arduino タグ）

§ 11. 最後に

実習を通して学生が得るもの

この実習を終えたとき、学生は以下を得られます：

知識・技能

- IoT システムの実装スキル
- プログラミングの基礎
- 電子回路の基礎
- デバッグ能力

思考力・判断力

- 理論と実践を結びつける力
- 問題を自分で解決する力
- システム全体を俯瞰する力

態度・姿勢

- 試行錯誤を楽しむ姿勢
- 協働して学ぶ態度
- 主体的に学ぶ習慣

これらは、IoT 技術だけでなく、あらゆる学習に役立つ力です。

IoT 技術応用
実習 1　デバイス制御
Part1: LED 制御の基礎　教員用指導ガイド

§ 1. 実習の概要

学習目標

知識面

- Arduino IDE の基本的な使い方を理解する
- Arduino と PC の接続方法を理解する
- LED の極性と抵抗の役割を理解する
- setup() と loop() の違いを理解する
- digitalWrite() と delay() の機能を理解する

技能面

- Arduino IDE をインストールできる
- ブレッドボードで基本的な回路を組める
- プログラムを書き込んで実行できる
- プログラムのパラメータを変更できる

態度面

- 試行錯誤を恐れず挑戦する
- 問題に直面した時、自分で解決策を考える
- 他の学生と協力して学習する

到達目標

最低限達成すべき目標：

- LED を正しく配線できる
- Blink プログラムを書き込んで実行できる
- LED が 1 秒ごとに点滅する

できれば達成したい目標：

- delay の値を変更して点滅速度を調整できる
- なぜその動作になるかを説明できる

§ 2. 事前準備

実習前に準備すること

機材の準備

- Arduino Uno（学生数 + 予備 3-5 個）
- USB ケーブル（学生数 + 予備 5 本）
- ブレッドボード（学生数分）

- LED 赤（学生数×3 + 予備 50 個）
- 抵抗 220Ω（学生数×3 + 予備 50 個）
- ジャンパーワイヤー（学生数×10 本 + 予備 50 本）

動作確認

- 予備を含む全ての Arduino の動作確認
- USB ケーブルがデータ転送対応か確認
- LED の極性と発光確認

PC 環境

- 実習室 PC の OS 確認（Windows 10/11 推奨）
- 管理者権限でのインストール可否確認
- インターネット接続確認
- Arduino.cc へのアクセス確認

配布資料

- 学生用ワークブック（人数分 + 予備）
- Arduino IDE インストール手順書（任意）
- 配線図

⚠️ **重要な確認事項**

- USB ポートの数が十分か（足りない場合は USB ハブ準備）
- ウイルス対策ソフトが Arduino IDE をブロックしないか
- ファイアウォールで Arduino.cc へのアクセスが可能か
- 予備 PC の準備（学生の PC 不調時用に 1-2 台）

§ 3. 指導の流れ

導入

説明内容

1. **注意事項**
    - 配線時は USB を抜く
    - 部品を丁寧に扱う

💡 **指導のコツ**

- 実際に完成品を見せると理解が早い
- 「失敗してもいい」という雰囲気作りが重要
- 質問しやすい環境を作る

ステップ 1: Arduino IDE インストール

指導のポイント

- ダウンロードに時間がかかることを予告
- 「Windows 版」を選ぶよう明確に指示
- インストール中に次のステップを説明

想定されるトラブル

問題	対処法
ダウンロードが進まない	ネットワーク確認、別ブラウザで試す
管理者権限エラー	IT 部門に連絡、または教員 PC で代行
起動しない	ウイルス対策ソフトの例外設定

ステップ 2: Arduino 接続と設定

指導のポイント

- USB ケーブルの向きを明確に説明
- 緑 LED の点灯を確認させる
- COM ポート番号はメモさせる

デモンストレーション

1. 実物を見せながら USB 接続
2. デバイスマネージャーの開き方を投影
3. Arduino IDE でのボード・ポート選択を実演

巡回時の確認項目

- 緑 LED が点灯しているか
- デバイスマネージャーに Arduino が表示されているか
- IDE で正しいボードとポートが選択されているか

Q&A: このステップでよくある質問

Q: COM ポートが表示されません

A:

1. USB ケーブルを抜き差ししてみる
2. 別の USB ポートに接続
3. ケーブルがデータ転送対応か確認
4. ドライバーの再インストール

ステップ 3: LED 配線（15 分）

指導のポイント

- 配線前に必ず USB を抜くよう徹底
- LED の極性を丁寧に説明
- ブレッドボードの構造を説明
- カラー配線図を見せながら説明

説明の順序

1. **ブレッドボードの構造**
    - 縦の列が内部で繋がっている

- 中央の溝で左右が分離

2. **LED の極性**
   - 長い足が+（アノード）
   - 短い足が−（カソード）
   - 実物を見せる
3. **抵抗の役割**
   - 電流を制限して LED を保護
   - 抵抗がないと LED が壊れる
4. **配線の手順**
   - ステップバイステップで説明
   - 各ステップで確認タイム

⚠️ **よくある配線ミス**

ミス	結果	見つけ方
LED の極性逆	光らない	長い足が D13 側か確認
抵抗なし	LED が壊れる可能性	回路に抵抗があるか確認
接続が浅い	不安定・動作しない	ワイヤーをしっかり差し直す
GND に未接続	光らない	回路が閉じているか確認

ステップ 4-5: プログラム書き込みと動作確認

指導のポイント

- Blink スケッチ例の開き方を実演
- コンパイルと書き込みの違いを説明
- 成功メッセージを確認させる

説明すべき内容

1. **コンパイル（検証）**
   - プログラムが正しいか確認
   - ✓ボタンをクリック
2. **書き込み**
   - Arduino にプログラムを転送
   - →ボタンをクリック
   - 数秒待つ
3. **動作確認**
   - LED が 1 秒ごとに点滅
   - 内蔵 LED も同時に点滅

Q&A: よくあるエラー

Q: "avrdude: stk500_recv()..." エラー

A: Arduino との通信エラー

1. ポート選択を確認
2. USB を抜き差し
3. 別の USB ポートを試す

Q: "Error opening serial port..." エラー

A: ポートが使用中

1. シリアルモニタを閉じる
2. Arduino IDE を再起動

ステップ 6: 実験とまとめ（25 分）

実験の指導

delay の値を変更して点滅速度を変える実験：

1. **delay(500) に変更**
    - 「どうなると思う？」と予想させる
    - 実際に試させる
    - 「なぜそうなったか」を考えさせる
2. **delay(2000) に変更**
    - 同様のプロセス
3. **自由に値を決めさせる**
    - 創造性を引き出す
    - 試行錯誤を楽しませる

まとめ（10 分）

1. **今回の振り返り**
    - 何ができるようになったか確認
    - setup() と loop() の違い
    - delay() の意味
2. **質疑応答**
    - 疑問点の解消
    - 次回の予告
3. **片付け**
    - 回路はそのまま（Part2 で使用）
    - ワークブックの提出確認

§ 4. 各ステップの詳細指導ポイント

Arduino IDE の説明

画面構成の説明

- **メニューバー**: ファイル操作、設定
- **ツールバー**: 検証、書き込みボタン

- **エディタ**: プログラムを書く場所
- **メッセージエリア**: エラーや成功メッセージ
- **コンソール**: 詳細情報

重要なメニュー

メニュー	内容
ファイル → スケッチ例	サンプルプログラム
ツール → ボード	Arduino の種類選択
ツール → シリアルポート	接続ポート選択
スケッチ → 検証・コンパイル	プログラムチェック
スケッチ → マイコンボードに書き込む	プログラム転送

Blink プログラムの解説

コード解説

```
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop() {
  digitalWrite(LED_BUILTIN, HIGH);
  delay(1000);
  digitalWrite(LED_BUILTIN, LOW);
  delay(1000);
}
```

各行の意味

コード	説明
`void setup()`	最初に 1 回だけ実行される初期設定
`pinMode(LED_BUILTIN, OUTPUT)`	内蔵 LED ピンを出力モードに設定
`void loop()`	繰り返し実行される処理
`digitalWrite(LED_BUILTIN, HIGH)`	LED を ON にする

delay(1000)	1000 ミリ秒（1 秒）待つ
digitalWrite(LED_BUILTIN, LOW)	LED を OFF にする

LED 回路の理論

LED の特性

- **順方向電圧**: 約 1.8-2.2V（赤色 LED）
- **順方向電流**: 10-20mA（標準）
- **極性**: アノード（+）、カソード（-）

抵抗値の計算（参考）

計算式: R = (Vs - Vf) / If

Vs: 電源電圧（5V）
Vf: LED 順方向電圧（約 2V）
If: LED 電流（20mA = 0.02A）

R = (5 - 2) / 0.02
= 3 / 0.02
= 150Ω

→ 220Ω を使用（安全マージン含む）

⚠ **学生への説明レベル**

初学者には詳細な計算は不要です。以下を理解させれば十分：

- 抵抗がないと LED が壊れる
- 220Ω は適切な抵抗値
- 極性を間違えると光らない（壊れない）

§ 5. トラブルシューティング

ハードウェアの問題

問題 1: LED が光らない

確認項目（優先順）：

1. **プログラムは正しく書き込まれているか？**
    - 書き込み成功メッセージを確認
    - 内蔵 LED（L）は点滅しているか？
2. **LED の極性は正しいか？**
    - 長い足が D13 側か確認
    - 逆の場合は差し直す

3. **配線は正しいか？**
    - D13 → LED 長い足
    - LED 短い足 → 抵抗
    - 抵抗 → GND
4. **接触不良はないか？**
    - ワイヤーを差し直す
    - LED を差し直す
5. **LED は壊れていないか？**
    - 別の LED に交換して試す

問題 2：Arduino が認識されない

確認手順：

1. **物理的な接続**
    - USB ケーブルを抜き差し
    - 別の USB ポートに接続
    - 緑 LED が点灯するか確認
2. **ケーブルの確認**
    - データ転送対応ケーブルか？
    - 充電専用ケーブルは不可
    - 別のケーブルで試す
3. **ドライバーの確認**
    - デバイスマネージャーを確認
    - 「不明なデバイス」はないか？
    - 必要に応じてドライバー再インストール
4. **Arduino 本体の確認**
    - 別の Arduino で試す
    - 予備機と交換

ソフトウェアの問題

問題 3：コンパイルエラー

エラーメッセージ	原因	対処法
expected ';'	セミコロン忘れ	該当行にセミコロンを追加
'xxx' was not declared	変数名のスペルミス	正しいスペルに修正
expected '}'	括弧の対応ミス	括弧の数を確認、対応を確認

問題 4：書き込みエラー

エラー	原因	対処法

avrdude: stk500_recv()	通信エラー	ポート確認、USB 抜き差し
Error opening serial port	ポート使用中	シリアルモニタを閉じる
Programmer not responding	ボード未選択	Arduino Uno を選択

§ 6. よくある質問と回答例

技術的な質問

Q1: なぜ抵抗が必要なのですか？

回答例:

LED は電流を制限しないと壊れてしまいます。抵抗は電流を適切な量に制限する役割があります。水道の蛇口を想像してください。抵抗は水の流れを調整する役割です。

補足（理解度が高い学生向け）:

Arduino のピンは 5V ですが、LED は約 2V しか必要としません。抵抗がないと過剰な電流が流れて LED が壊れます。220Ω の抵抗で約 15mA の電流に制限しています。

Q2: LED_BUILTIN って何ですか？

回答例:

Arduino 基板に最初から付いている小さな LED（L と書いてあるもの）のことです。プログラムでは 13 番ピンと同じ意味です。今回は外付け LED も D13 に接続したので、同時に光ります。

Q3: delay(1000) の 1000 って何ですか？

回答例:

1000 ミリ秒のことです。1000 ミリ秒 = 1 秒です。delay(500)なら 0.5 秒、delay(2000)なら 2 秒待ちます。

確認質問で理解を深める:

「では、0.3 秒待ちたい場合は？」→ delay(300)

Q4: なぜ loop() は繰り返されるのですか？

回答例:

Arduino の仕組みがそうなっています。setup()が終わると、自動的に loop()を何度も繰り返し実行します。これが Arduino の特徴です。

イメージ図:

```
起動 → setup() → loop() → loop() → loop() → ...
        ↑         ↓       ↓       ↓
        1回      永遠に繰り返す
```

実習に関する質問

Q5: 間違えて LED を逆に付けたらどうなりますか？

回答例:

光らないだけで、壊れません。LED は極性を間違えると電流が流れない性質があります。だから安心して試行錯誤してください。

Q6：別の色の LED も使えますか？

回答例：

はい、使えます！緑、青、黄色など、いろいろな色の LED があります。ただし、青や白の LED は電圧が少し高いので、抵抗値を変える必要があります。今日は赤を使いましょう。

Q7：家でもできますか？

回答例：

できます！Arduino Uno は数千円で購入できます。興味があれば、インターネットで「Arduino 入門キット」を検索してみてください。今日使った部品が全て入ったセットがあります。

理解を深める質問（教員から学生へ）

💡 **学生の理解度を確認する質問例**

- 「setup()と loop()の違いは？」
- 「HIGH と LOW は何を意味する？」
- 「delay(500)は何秒？」
- 「抵抗がないとどうなる？」
- 「LED の長い足はどっち側？」

§ 7. 発展的な内容

早く終わった学生への課題

レベル 1：点滅パターンの変更

以下のパターンを作ってみよう：

- 短く点滅（0.1 秒 ON、0.1 秒 OFF）
- ゆっくり点滅（2 秒 ON、2 秒 OFF）
- モールス信号風（短・短・短・長・長・長・短・短・短）

レベル 2：複数の LED を使う

2 つの LED を交互に点滅させる：

- D12 にも LED を追加
- D13 と D12 が交互に光るプログラム

レベル 3：パターンを考える

オリジナルの点滅パターンを作ってみよう：

- リズムを考える
- 何かのメッセージを表現する

次回（Part2）への準備

学生に伝えること

- Part1 の回路はそのまま残しておく
- 次回はセンサーを追加する

- 温度と湿度を測定する
- より実用的なシステムになる

教員側の準備

- DHT11 センサーを人数分準備
- 追加のジャンパーワイヤー準備
- Part2 のワークブック印刷

§ 参考資料

推奨 Web サイト

- Arduino 公式サイト: https://www.arduino.cc/
- Arduino 日本語リファレンス
- Arduino スケッチ例集

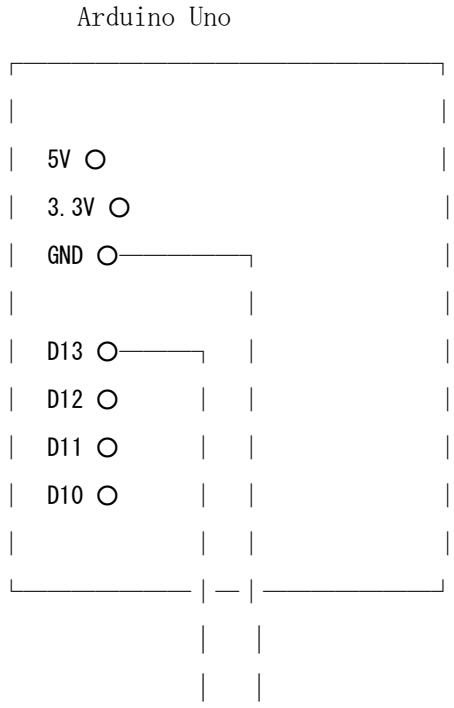
推奨書籍

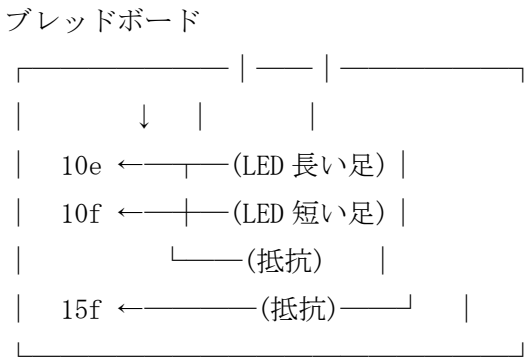
- 「Arduino をはじめよう」(オライリー・ジャパン)
- 「Arduino で計る・測る」(CQ 出版社)
- 「たのしくできる Arduino 電子工作」(東京電機大学出版局)

§ 付録

配線図（拡大版）

Arduino Uno 配線図





部品リスト:
- Arduino Uno: 1 個
- LED（赤）: 1 個
- 抵抗 220Ω: 1 個
- ジャンパーワイヤー（赤）: 1 本
- ジャンパーワイヤー（黒）: 1 本

チェックリスト

実習開始前チェック

項目	チェック
機材の配置完了	☐
Arduino 動作確認	☐
配布物の準備	☐
予備部品の準備	☐

IoT 技術応用
実習１ デバイス制御
Part2: センサーでデータ取得 教員用指導ガイド

§ 1. 実習の概要

学習目標

知識面

- ライブラリの役割と利用方法を理解する
- DHT11 センサーの仕組みと用途を理解する
- シリアル通信の基本を理解する
- デジタルセンサーのデータ取得方法を理解する
- float 型の意味と nan（Not a Number）を理解する

技能面

- Arduino IDE でライブラリをインストールできる
- センサーを正しく配線できる
- センサーデータを取得するプログラムを書ける
- シリアルモニタを使ってデータを確認できる
- センサーエラーを検出・対処できる

態度面

- センサーの特性を観察・理解しようとする
- 実験を通じて科学的な思考を養う
- 測定値の妥当性を判断する

Part1 からの発展

Part1 との関連性

Part1 では：

- 出力（LED 制御）を学んだ
- digitalWrite() で信号を送った
- delay() で時間を制御した

Part2 では：

- 入力（センサーデータ取得）を学ぶ
- readTemperature() / readHumidity() でデータを受け取る
- Serial.print() でデータを表示する

→ 入力と出力が組み合わさって、IoT システムが完成する

§ 2. 事前準備

実習前に準備すること

機材の準備

- DHT11 センサー（学生数 + 予備 10 個）
- ジャンパーワイヤー（学生数×5 本 + 予備 50 本）
- Part1 で使用した Arduino 回路（そのまま）

DHT11 センサーの事前確認

- 全センサーの動作確認（重要！）
- ピン配置の確認（モジュール型か生センサーか）
- 不良品の除外
- 測定精度の確認（室温で±2℃程度）

⚠️ DHT11 の品質について

DHT11 は低価格センサーのため、以下の問題が発生することがあります：

- 初期不良（5-10%程度）
- 測定値のバラツキ（±2℃程度）
- 応答の遅れ（2 秒以上必要）
- 湿度測定の不安定性

対策：予備を多めに準備し、事前に全数動作確認を実施

DHT11 のピン配置確認

2 種類の DHT11

タイプ	特徴	ピン配置
生センサー	青色の四角い本体、4 本ピン	1:VCC, 2:DATA, 3:NC, 4:GND
モジュール型	基板付き、3 本ピン	1:VCC, 2:DATA, 3:GND（基板に印刷）

注意：購入したセンサーがどちらのタイプか事前に確認し、指導内容を調整してください。

ソフトウェアの準備

ライブラリの事前確認

- 実習室 PC でライブラリマネージャーが使えるか確認
- インターネット接続が必要
- ファイアウォールでブロックされないか確認

オフライン環境の場合

インターネット接続がない場合の対策：

1. 事前にライブラリファイル（.zip）をダウンロード
2. ネットワーク共有フォルダに配置
3. 「ZIP 形式のライブラリをインストール」で対応

必要なライブラリ：

- DHT sensor library by Adafruit
- Adafruit Unified Sensor

配布資料の準備

- Part2 学生用ワークブック（人数分 + 予備）
- DHT11 配線図（カラー印刷推奨）
- プログラムコード（印刷または投影用）

§ 3. 指導の流れ

ステップ 1: ライブラリインストール

指導のポイント

- ライブラリの概念を丁寧に説明
- 「便利な道具セット」に例える
- 2 つのライブラリが必要なことを明示

説明の順序

1. **ライブラリとは**
    - 「誰かが作った便利な機能のセット」
    - 「料理で言えば、調味料セット」
    - 「一から作る必要がない」
2. **インストール手順を実演**
    - ライブラリマネージャーを開く
    - 検索して見つける
    - インストールボタンをクリック
3. **2 つ目のライブラリも同様に**
    - 「依存関係」の説明は省略して OK
    - 「2 つセットで必要」とだけ伝える

⚠️ **よくあるトラブル**

問題	原因	対処法
ライブラリが見つからない	ネットワーク問題	接続確認、再起動
インストールできない	権限不足	管理者権限で実行
「すでにインストール済み」	前回の残り	問題なし、次へ進む

ステップ 2-3: センサー確認と配線

指導のポイント

- センサーの向きを明確に示す
- ピン配置を丁寧に説明
- 配線の色分けを推奨

- Part1 の回路はそのままで OK

センサーの観察（5 分）

1. **実物を見せる**
    - 正面と背面を見せる
    - 格子模様がある面が正面
    - ピンの位置を確認
2. **ピン配置の説明**
3.
4. DHT11 ピン配置（正面から見た図）
5.
6. ┌─────────────────┐
7. │ DHT11 │
8. │ ┌─────┐ │
9. │ │▓▓▓▓▓│ │
10. │ │▓▓▓▓▓│ │
11. │ └─────┘ │
12. │ │ │ │ │
13. │ 123 │
14. └─────────────────┘
15.
16. **1: VCC (5V)**
17. **2: DATA (D2)**
18. **3: GND**

19. **配線の色分け推奨**
    - VCC → 赤ワイヤー
    - DATA → 黄ワイヤー
    - GND → 黒ワイヤー

⚠️ 配線ミスの早期発見

以下の点を重点的にチェック：

- VCC が 5V に接続されているか（3.3V ではない）
- DATA が D2 に接続されているか
- GND が GND に接続されているか
- センサーの向きは正しいか
- ワイヤーがしっかり差さっているか

ステップ 4: プログラム作成

指導のポイント

- プログラムの全体像を先に説明
- 各行の意味を説明しながら入力

- タイプミスに注意させる

プログラムの説明順序

全体構造の説明

```
1.  #include           // ライブラリを使う宣言
2.  #define DHTPIN 2           // センサーは D2 に接続
3.  #define DHTTYPE DHT11      // センサーの種類
4.
5.  DHT dht(DHTPIN, DHTTYPE); // センサーを使う準備
6.
7.  void setup() {
8.    Serial.begin(9600);      // シリアル通信開始
9.    dht.begin();             // センサー初期化
10. }
11.
12. void loop() {
13.   float temp = dht.readTemperature(); // 温度取得
14.   float humi = dht.readHumidity();    // 湿度取得
15.
16.   Serial.print("温度: ");
17.   Serial.print(temp);
18.   Serial.println("℃");
19.
20.   Serial.print("湿度: ");
21.   Serial.print(humi);
22.   Serial.println("%");
23.
24.   delay(2000);             // 2 秒待つ
}
```

重要な行の解説

コード	説明（学生向け）
#include <DHT.h>	ライブラリを読み込む
#define DHTPIN 2	センサーは D2 ピンだ、と教える
Serial.begin(9600)	パソコンとの会話を始める
dht.readTemperature()	センサーに温度を聞く

Serial.print()	パソコンに結果を送る

入力時の注意点（10 分）

- 大文字・小文字を正確に
- セミコロン（;）を忘れずに
- 括弧の対応を確認
- 全角スペースに注意

💡 **プログラム入力のサポート**

- 投影してゆっくり入力を進める
- 5-10 行ごとに確認タイムを設ける
- 早く終わった学生に近くの学生を手伝わせる
- コピー&ペーストは推奨しない（学習効果のため）

ステップ 5: 動作確認と実験

シリアルモニタの説明

1. **シリアルモニタとは**
   - 「Arduino からのメッセージを見る窓」
   - 「温度・湿度がここに表示される」
2. **開き方の実演**
   - ツール → シリアルモニタ
   - または Ctrl+Shift+M
   - ボーレート 9600 に設定
3. **表示内容の確認**
   - 温度と湿度が 2 秒ごとに更新
   - 値が妥当か判断させる
   - 室温として合理的か？

実験の指導

1. **実験 1: 室温測定**
   - 現在の室温を記録
   - 隣の人と比較
   - なぜ微妙に違うか考える
2. **実験 2: 息を吹きかける**
   - 温度と湿度の変化を観察
   - 「なぜ両方上がる？」と質問
   - 人間の呼気の特性を説明
3. **実験 3: 手で温める**
   - 温度上昇を確認
   - 体温との関係を考える

実験での気づきを促す質問

- 「隣の人と温度が違うのはなぜ？」

- 「息を吹きかけると湿度も上がるのはなぜ？」
- 「センサーの反応は速い？遅い？」
- 「どこで使えそう？」

§ まとめ

振り返りの進め方

1. **今日学んだこ**
    - ライブラリの使い方
    - センサーデータの取得
    - シリアル通信
2. **重要な概念**
    - 入力と出力
    - float 型と nan
    - Serial.print() と println() の違い
3. **次回の予告**
    - Part3 で LED とセンサーを組み合わせる
    - 「温度が高いと LED が光る」など
4. **質疑応答**

💡 **まとめのポイント**

- Part1 と Part2 で「入力」と「出力」が揃った
- これで IoT の基本が完成した
- 次は組み合わせて応用する

§ 4. 各ステップの詳細指導ポイント

ライブラリの概念説明

初学者向けの説明例

例え話 1：料理

「DHT11 を使うのは、料理で調味料セットを使うようなものです。一から作らなくても、誰かが作った便利なものを使えます。」

例え話 2：アプリ

「スマホでアプリをインストールするのと同じです。DHT11 を使うための『アプリ』を Arduino IDE にインストールします。」

依存関係の説明（簡易版）

「2 つのライブラリが必要です。これは、ゲームをするのにゲーム本体とコントローラーが両方必要なのと似ています。」

シリアル通信の説明

Serial.begin(9600) の意味

- 「Arduino と PC が会話する準備」

- 「9600 は会話のスピード（ボーレート）」
- 「両側で同じスピードに設定する必要がある」

Serial.print() と Serial.println() の違い

関数	動作	例
Serial.print()	改行しない	「温度：25℃」が横に続く
Serial.println()	改行する	「温度：25℃」の後に改行

板書用の図：

```
Serial.print("温度：");
Serial.print(25);
Serial.println("℃");
```

表示結果：
温度：25℃
↓（ここで改行）

float 型と nan の説明

float 型とは

- 「小数点のある数字を扱う型」
- 「25.5 や 60.3 のような値」
- 「int だと整数しか扱えない」

nan とは（Not a Number）

- 「数字ではない」という意味
- 「センサーエラーのサイン」
- 「センサーが正しく読めなかった」

nan が出る原因

1. 配線ミス（最も多い）
2. センサーの不良
3. 電源不足
4. 初回読み取りの失敗（数秒で解消）

§ 5. DHT11 センサーの技術情報

基本仕様

DHT11 の仕様

項目	仕様

電源電圧	3.3V – 5.5V
温度測定範囲	0℃ – 50℃
温度測定精度	±2℃
湿度測定範囲	20% – 90% RH
湿度測定精度	±5% RH
サンプリング周期	1 秒に 1 回（最大）
応答時間	6-15 秒（湿度）

通信プロトコル

DHT11 の通信方式

DHT11 は独自のシングルワイヤー通信プロトコルを使用：

- 1 本のデータ線で双方向通信
- データは 40 ビット（5 バイト）
- 湿度整数部、湿度小数部、温度整数部、温度小数部、チェックサム

注意：この詳細は学生には説明不要。ライブラリが全て処理してくれます。

測定精度と限界

⚠️ DHT11 の限界

- **低精度：**±2℃、±5%RH の誤差
- **低分解能：**整数値のみ（小数点なし）
- **遅い応答：**2 秒以上待つ必要
- **範囲制限：**0-50℃、20-90%のみ

上位機種との比較

センサー	温度精度	湿度精度	用途
DHT11	±2℃	±5%	学習用・簡易測定
DHT22	±0.5℃	±2%	より高精度が必要な場合
BME280	±1℃	±3%	温湿度＋気圧測定

§ 6. トラブルシューティング

問題 1：nan が表示される

最も多いトラブル

シリアルモニタに「nan」と表示される場合の対処法：

確認手順（優先順）

1. **数秒待つ**（1 分）
    - 初回読み取りは失敗することがある
    - 2-3 回更新を待つ
    - それでも nan なら次へ
2. **配線確認**（5 分）
    - VCC は 5V に接続？
    - DATA は D2 に接続？
    - GND は GND に接続？
    - 接触不良はない？
3. **プログラム確認**（3 分）
    - #define DHTPIN 2 は正しい？
    - #define DHTTYPE DHT11 は正しい？
    - タイプミスはない？
4. **センサー交換**（2 分）
    - 予備のセンサーと交換
    - センサーの初期不良の可能性
5. **Arduino 再起動**（1 分）
    - USB を抜き差し
    - プログラムを再書き込み

問題 2：温度が異常値

異常値の例と対処

表示値	原因	対処法
0℃	センサーエラー	配線確認、センサー交換
50℃以上	センサー不良	センサー交換
マイナス温度	プログラムエラー	コード確認、再書き込み
値が変わらない	loop()が動いていない	delay 確認、再起動

問題 3：ライブラリが見つからない

対処手順

1. **インターネット接続確認**
    - ブラウザで接続確認
    - ファイアウォール確認

2. **ライブラリマネージャー再起動**
    - 一度閉じて開き直す
    - Arduino IDE を再起動
3. **手動インストール**
    - GitHub から ZIP ダウンロード
    - ZIP 形式のライブラリをインストール

問題 4：シリアルモニタに何も表示されない

確認項目

- ボーレートが 9600 になっているか
- 正しい COM ポートに接続しているか
- Serial.begin(9600)が setup()に書かれているか
- プログラムは正しく書き込まれたか
- USB ケーブルはデータ転送対応か

問題 5：コンパイルエラー

よくあるエラーと対処

エラーメッセージ	原因	対処法
'DHT' does not name a type	ライブラリ未インストール	ライブラリをインストール
expected ';' before...	セミコロン忘れ	該当行にセミコロン追加
'DHTPIN' was not declared	#define 忘れ	#define DHTPIN 2 を追加
expected ')' before...	括弧の対応ミス	括弧の数を確認

§ 7. よくある質問と回答例

技術的な質問

Q1：なぜライブラリが必要なのですか？

回答例：

DHT11 は複雑な通信方式を使っています。それを一から書くのは大変なので、誰かが作った便利なライブラリを使います。これにより、簡単なコードで温度や湿度を読み取れます。

例え：

「車を運転するのに、エンジンの仕組みを全部知る必要はないですよね？ハンドルとアクセルの使い方だけ知っていれば運転できます。ライブラリも同じです。」

Q2：Serial.begin(9600) の 9600 は何ですか？

回答例：

ボーレート（通信速度）です。1 秒間に 9600 ビットのデータを送る速度です。Arduino と PC の両方を

同じ速度に設定する必要があります。

例え：

「2 人で会話するとき、同じ言語・同じ速度で話さないと通じませんよね？それと同じです。」

Q3：float って何ですか？

回答例：

小数点のある数字を扱うためのデータ型です。温度は「25.5℃」のように小数点以下も必要なので、float 型を使います。int 型だと整数しか扱えません。

型	扱える値	例
int	整数のみ	25, 26, 27
float	小数点あり	25.5, 26.3, 27.8

Q4：なぜ delay(2000) が必要なのですか？

回答例：

DHT11 は測定に時間がかかるセンサーです。最低 1 秒、推奨 2 秒待つ必要があります。早すぎると正しい値が読めません。

実験で示す：

delay(100)に変更すると、エラーが出たり不安定になることを見せる

Q5：nan って何ですか？

回答例：

Not a Number（数字ではない）の略です。センサーが正しく読み取れなかった時に表示されます。配線ミスやセンサーエラーの可能性があります。

対処法を教える：

1. まず数秒待つ
2. 配線を確認
3. それでもダメなら教員に相談

Q6：温度が隣の人と違うのはなぜですか？

回答例：

いくつか理由があります：

- センサーの個体差（±2℃の誤差）
- 設置場所の違い（窓側、エアコンの近くなど）
- PC の熱の影響

これは正常です。DHT11 は学習用の安価なセンサーなので、多少のバラツキはあります。

応用的な質問

Q7：このセンサーはどこで使われていますか？

回答例：

いろいろな場所で使われています：

- エアコンの温湿度センサー

- 気象観測装置
- 植物の栽培管理
- 冷蔵庫や食品保管
- 博物館の環境管理

ただし、DHT11 は精度が低いので、本格的な用途には DHT22 や BME280 などのより高精度なセンサーが使われます。

Q8：他にどんなセンサーがありますか？

回答例:

たくさんあります！

- 光センサー（明るさ）
- 距離センサー（超音波、赤外線）
- 傾きセンサー（加速度センサー）
- ガスセンサー（CO2、アルコールなど）
- 圧力センサー
- 音センサー

次回以降で他のセンサーも使っていきます！

§ 8. 発展的な内容

早く終わった学生への課題

レベル 1：表示形式の変更

以下の表示形式に変更してみよう：

- 「Temperature: 25.0C / Humidity: 60.0%」形式
- 表を使った表示形式
- 華氏（°F）での表示

華氏変換の式：

```
float fahrenheit = (temp * 9.0 / 5.0) + 32.0;
```

レベル 2：条件分岐を追加

温度に応じてメッセージを表示：

```
if (temp < 20) {
  Serial.println("寒いです");
} else if (temp > 28) {
  Serial.println("暑いです");
} else {
  Serial.println("快適です");
}
```

レベル 3：不快指数の計算

温度と湿度から不快指数を計算：

```
float discomfort = 0.81 * temp + 0.01 * humi * (0.99 * temp - 14.3) + 46.3;
Serial.print("不快指数: ");
Serial.println(discomfort);
```

- 55 未満：寒い
- 55-60：肌寒い
- 60-65：何も感じない
- 65-70：快い
- 70-75：暑くない
- 75-80：やや暑い
- 80-85：暑くて汗が出る
- 85 以上：暑くてたまらない

Part3 への準備

学生に伝えること

- Part1 と Part2 の回路はそのまま残す
- 次回は入力と出力を組み合わせる
- 「温度が高いと LED が光る」などの制御
- より実用的な IoT システムになる

教員側の準備

- 次回も同じ機材を使用
- 追加部品は不要（Part1+Part2 の組み合わせ）
- Part3 のワークブック印刷

補足教材の紹介

推奨 Web サイト

- Adafruit DHT sensor library GitHub
- Arduino 公式チュートリアル
- Arduino センサー活用事例集

推奨動画

- DHT11 の使い方（YouTube 検索）
- シリアル通信の仕組み
- 温湿度センサーの応用例

発展的な学習テーマ

- DHT11 と DHT22 の比較実験
- 複数センサーの同時使用
- データの SD カード保存
- LCD ディスプレイへの表示

§ 指導のまとめ

Part2 指導の重要ポイント

- **センサーの事前確認**：不良品を早期発見
- **ライブラリの重要性**：概念を丁寧に説明
- **nan への対処**：配線確認を最優先

- **測定値の妥当性**：科学的思考を促す
- **実験を楽しむ**：センサーの特性を体験
- **Part1 との関連**：入力と出力の組み合わせを強調

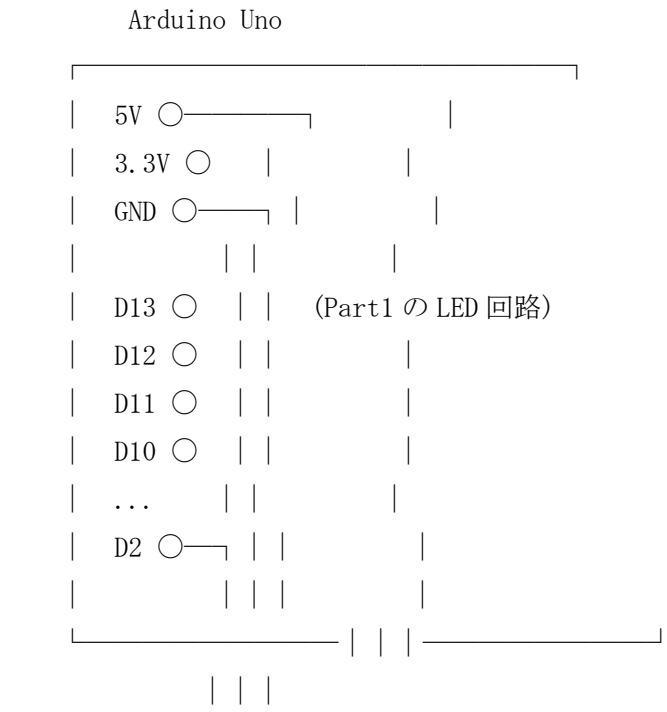
💡 **Part2 授業を成功させるコツ**

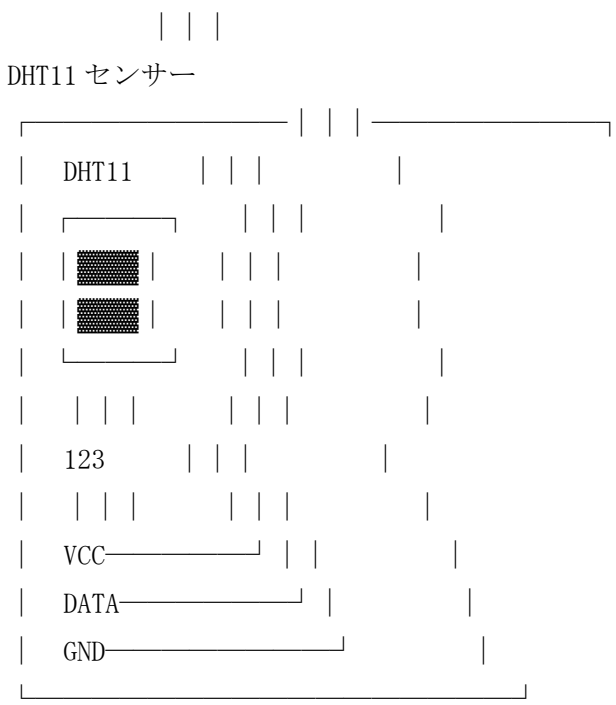
1. **センサーの事前テスト**
    - 全センサーの動作確認を必ず実施
    - 不良品を除外しておく
    - 予備を十分に準備（10 個以上）
2. **ライブラリインストールの時間配分**
    - ダウンロードに時間がかかることを想定
    - オフライン対策も準備
3. **nan トラブルへの対応**
    - 最も多いトラブルなので時間を確保
    - 配線確認の手順を明確に
    - センサー交換をスムーズに
4. **実験の楽しさ**
    - 息を吹きかける実験は盛り上がる
    - 「科学者になった気分」を演出
    - 測定値の変化を観察させる

§ 付録

配線図（拡大版）

Arduino Uno + DHT11 配線図





配線リスト:

1. DHT11 VCC → Arduino 5V（赤ワイヤー）
2. DHT11 DATA → Arduino D2（黄ワイヤー）
3. DHT11 GND → Arduino GND（黒ワイヤー）

※Part1 の LED 回路はそのまま残す

プログラム全文（印刷用）

```
// DHT11 温湿度センサー プログラム

#include <DHT.h>

#define DHTPIN 2
#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  dht.begin();
}

void loop() {
  float temp = dht.readTemperature();
```

```
float humi = dht.readHumidity();

Serial.print("温度: ");
Serial.print(temp);
Serial.println("℃");

Serial.print("湿度: ");
Serial.print(humi);
Serial.println("%");

delay(2000);
```

IoT 技術応用
実習１ デバイス制御
Part3: 条件による制御 教員用指導ガイド

§ 1. 授業概要

授業の位置づけ

Part3 は、Part1 と Part2 で学んだ「出力」と「入力」を統合する重要な回です。

- **Part1**：出力（LED の制御）
- **Part2**：入力（センサーデータの取得）
- **Part3**：統合（条件による制御）← 今回

今回の授業で、学生は初めて「IoT システム」の全体像を理解します。

学習目標

知識面の目標

- if 文の基本構造と動作原理を理解する
- 比較演算子の種類と使い方を理解する
- 条件分岐の順序の重要性を理解する
- IoT システムの基本構造（入力→判断→出力）を理解する

技能面の目標

- if、else if、else を正しく書ける
- センサー値を条件として使える
- 閾値を調整してシステムの動作を変更できる
- 簡単な IoT システムを設計・実装できる

項目	内容
導入	振り返り、目標説明、デモ
ステップ 1：if 文の基本	if 文の構造、比較演算子、例題
ステップ 2：else if/else	複数条件の判定、使い方
ステップ 3：プログラム作成	プログラム入力、検証
ステップ 4：動作確認	書き込み、初期確認

ステップ 5：実験	手で温める、閾値変更
まとめ	振り返り、次回予告

必要な機材

学生 1 人あたり

- Arduino Uno（Part1 の回路そのまま）
- DHT11 センサー（Part2 の回路そのまま）
- USB-A to B ケーブル
- PC（Arduino IDE インストール済み）
- ワークブック

教員用

- デモ用 Arduino（完成品）
- プロジェクター/大型ディスプレイ
- ホワイトボード/黒板
- 予備のセンサー（5 個程度）

💡 **重要：**Part1 と Part2 の回路をそのまま使用します。新しい配線は不要です！

§ 2. 授業準備

デモ用システムの準備

授業の冒頭で実際に動作するシステムを見せることが重要です。

デモの内容

1. 室温での状態（おそらく消灯または緩やかな点滅）
2. 手で温めると点滅が始まる様子
3. さらに温めると点滅が速くなる様子
4. シリアルモニタでの温度と状態の表示

デモのポイント

- シリアルモニタを大きく表示
- LED の点滅が見やすいようにカメラで拡大
- 温度の変化と動作の変化を対応させて説明
- 「これを今日、皆さんが作ります」と伝える

板書補足

補足事項について板書する際の内容

if 文の基本構造

```
if（条件 1）{
    処理 1
} else if（条件 2）{
    処理 2
} else {
```

```
    処理3
}
```

比較演算子

== ： 等しい

!= ： 等しくない

> ： より大きい

>= ： 以上

< ： より小さい

<= ： 以下

フローチャート

温度測定

↓

28度以上? → YES → 速く点滅

↓ NO

25度以上? → YES → ゆっくり点滅

↓ NO

消灯

§ 3. 導入

これまでの振り返り

話す内容

- Part1では「出力」を学んだ（LEDを光らせる）
- Part2では「入力」を学んだ（温度を測る）
- 今日はこの2つを組み合わせる

学生への質問例

- 「Part1で何をしたか？」→「LEDを点滅させた」
- 「Part2では？」→「温度と湿度を測った」
- 「今日は何をすると思うか？」→「組み合わせる」

今日の目標説明

デモンストレーション

実際に動作するシステムを見せながら説明します。

1. 「今、室温は23度です。LEDは消えています」
2. 「手で温めます。25度を超えると...」
3. 「LEDが点滅し始めました！」
4. 「さらに温めて28度を超えると...」
5. 「点滅が速くなりました！」

強調ポイント：

- 温度という「条件」によって動作が変わる
- これがIoTシステムの基本
- エアコンも冷蔵庫も同じ仕組み

if 文の紹介

日常生活との関連付け

if 文を日常の例で説明すると理解しやすくなる。

- 「もし雨なら、傘を持っていく」
- 「もし寒いなら、上着を着る」
- 「もしお腹が空いたら、ご飯を食べる」

プログラムでも同じ：

- 「もし温度が 25 度以上なら、LED を点滅させる」

板書

必要に応じて if 文の基本構造を板書して、全員で確認する。

⚠ **導入での注意点**

- デモが失敗しないよう、事前に十分テストする
- 温度の変化には時間がかかるので、予熱しておく
- シリアルモニタの文字サイズを大きくしておく
- 学生が「面白そう」と感じられるような雰囲気作り

§ 4. ステップ 1：if 文の基本

if 文の構造説明

教え方のポイント

抽象的な説明だけでは理解しにくいので、具体例を多く使う。

説明の流れ

1. **基本形を見せる**
2. if（条件）{
3. 実行する処理

}

4. **日本語で説明**

「もし（条件が正しければ）{ この中を実行する }」

5. **具体例を示す**
6. if (temp >= 25) {
7. digitalWrite(13, HIGH);

}

「もし温度が 25 度以上なら、D13 を HIGH にする」

板書のポイント

- 括弧の種類を色分け（丸括弧、中括弧）
- 条件部分と処理部分を明確に区別
- インデント（字下げ）を強調

比較演算子の説明

覚えるべき演算子

演算子	読み方	意味	使用例
==	イコール イコール	等しい	temp == 25
!=	ノット イコール	等しくない	temp != 25
>	大なり	より大きい	temp > 25
>=	大なり イコール	以上	temp >= 25
<	小なり	より小さい	temp < 25
<=	小なり イコール	以下	temp <= 25

強調すべき点

== と = の違い

- ==（イコール 2 つ）：比較「等しいかどうか」
- =（イコール 1 つ）：代入「値を入れる」

if 文の中では必ず == を使う！

教え方のコツ

- 「以上」と「より大きい」の違いを数直線で説明
- 具体的な数字で例を示す（temp = 30 のとき...）
- 学生に口頭で答えさせる（クイズ形式）

簡単な例題

クイズ形式で進める

「温度が 30 度のとき、次の条件は〇か×か？」

条件	答え	説明
temp >= 25	○	30 は 25 以上
temp < 25	×	30 は 25 より小さくない
temp == 30	○	30 は 30 に等しい
temp > 30	×	30 は 30 より大きくない（30 ちょうど）

進め方

1. 問題を投影する
2. 10 秒考える時間を与える

3. 挙手で答えさせる
4. 正解を説明する

💡 **ポイント：**間違えた学生を責めない。「良い質問です」「一緒に考えましょう」など、ポジティブな雰囲気を作る。

§ 5. ステップ 2：else if と else（10 分）

else if の説明

なぜ else if が必要か

if だけでは 1 つの条件しか判定できない。今日は 3 段階に分けたい：

- 25 度未満：消灯
- 25 度以上 28 度未満：ゆっくり点滅
- 28 度以上：速く点滅

フローチャートで説明

ホワイトボードにフローチャートを描きながら説明すると効果的。

温度を測る
　↓
28 度以上？ → YES → 速く点滅 → 終了
　↓ NO
25 度以上？ → YES → ゆっくり点滅 → 終了
　↓ NO
　消灯 → 終了

強調ポイント：

- 上から順番にチェックされる
- 最初に当てはまった条件だけが実行される
- 1 つ実行したら、残りはスキップされる

else の説明

else の役割

「それ以外の場合」を表す

```
if (temp >= 28) {
    速く点滅
} else if (temp >= 25) {
    ゆっくり点滅
} else {
    消灯  // それ以外（25 度未満）
}
```

教え方のポイント

- else には条件を書かない
- 「それ以外全部」という意味
- 必ずしも else は必要ない（if だけでも可）

- でも、今回は「何もしない」も明示的に書く

順序の重要性

⚠️ 順序を間違えるとどうなるか

間違った例：

```
if (temp >= 25) {
    ゆっくり点滅
} else if (temp >= 28) {
    速く点滅  // この条件は永遠に実行されない！
}
```

なぜダメか：

温度が 30 度のとき、最初の条件（temp >= 25）に当てはまってしまい、ゆっくり点滅してしまう。2 番目の条件はチェックされない。

正しい順序：

厳しい条件（大きい数）を先に書く！

学生への説明

「上から順番に、最初に当てはまった条件だけが実行される。だから、厳しい条件を先に書かないと、永遠に実行されない条件ができてしまう」

💡 理解度チェック

「温度が 26 度のとき、どの処理が実行されるか？」

挙手で答えさせて、理解度を確認する。

§ 6. ステップ 3：プログラム作成（30 分）

プログラムの全体像説明

Part2 からの変更点を説明

Part2 のプログラムに、以下を追加する：

- #define LEDPIN 13 - LED ピンの定義
- pinMode(LEDPIN, OUTPUT) - setup()内
- if 文 - loop()内、温度表示の後

プログラムの流れを説明

1. 温度と湿度を読み取る（Part2 と同じ）
2. シリアルモニタに表示（Part2 と同じ）
3. 温度によって動作を変える（新規）
4. 2 秒待つ（Part2 と同じ）

プログラム入力

入力の進め方

プログラムを投影しながら、ゆっくり入力させる。

内容	確認ポイント

ライブラリと define	LEDPIN 13 の追加
setup 関数	pinMode 追加
loop 関数（if 文まで）	温度・湿度表示（Part2 と同じ）
if 文の入力	括弧、インデント、条件

入力時の注意点

頻出するミス：

- セミコロン（;）忘れ
- 括弧の対応ミス（{ と } の数）
- else if を elseif と書く（スペース必要）
- 比較演算子のミス（>= を => と書くなど）
- インデント（字下げ）忘れ

対策：

- 5 行ごとに全員が追いついているか確認
- 括弧の色が変わることを説明（IDE 機能）
- インデントは Tab キーで簡単にできることを伝える

詳細な入力手順

1. define 文の追加

```
#define DHTTYPE DHT11
```

この下に追加：

```
#define LEDPIN 13
```

2. setup 関数の修正

```
dht.begin();
```

この下に追加：

```
pinMode(LEDPIN, OUTPUT);
```

3. loop 関数の if 文

温度・湿度表示の後、delay(2000)の前に追加：

```
if (temp >= 28) {
    Serial.println("状態: 高温・速く点滅");
    digitalWrite(LEDPIN, HIGH);
    delay(200);
    digitalWrite(LEDPIN, LOW);
    delay(200);
} else if (temp >= 25) {
    Serial.println("状態: やや高温・ゆっくり点滅");
    digitalWrite(LEDPIN, HIGH);
```

```
    delay(1000);
    digitalWrite(LEDPIN, LOW);
    delay(1000);
} else {
    Serial.println("状態: 通常・消灯");
    digitalWrite(LEDPIN, LOW);
}
```

各部分の説明

- if (temp >= 28) - 28 度以上なら
- Serial.println(...) - 状態を表示
- digitalWrite(LEDPIN, HIGH); delay(200); LOW; delay(200);- 0.2 秒ごとに点滅
- else if (temp >= 25) - 25 度以上 28 度未満なら
- delay(1000) - 1 秒ごとに点滅
- else - 25 度未満なら
- digitalWrite(LEDPIN, LOW) - 消灯

プログラムの確認

検証の手順

1. 全員が入力を終えたか確認
2. チェックマーク（✓）をクリックさせる
3. エラーが出た学生を確認
4. 巡回しながら個別対応

よくあるエラーと対処法

エラーメッセージ	原因	対処法
expected ';' before ...	セミコロン忘れ	エラー行の前の行を確認
expected '}' at end of input	括弧の閉じ忘れ	{ と } の数を数える
'elseif' was not declared	else if を elseif と書いている	スペースを入れる
lvalue required as left operand	== を = と書いている	比較は == を使う

全員が検証完了したら

「コンパイルが完了しました」と表示された人は、隣の人を手伝うように促す。

§ 7. ステップ 4：動作確認

プログラムの書き込み

手順

1. 矢印（→）をクリックして書き込む
2. 「マイコンボードへの書き込みが完了しました」を確認
3. シリアルモニタを開く

注意点

- 書き込み中は USB ケーブルを抜かない
- 書き込みには 15-20 秒かかる
- Arduino のオレンジ色 LED が点滅する（正常）

シリアルモニタで確認

確認項目

- 温度が表示されているか
- 湿度が表示されているか
- 状態が表示されているか
- LED の状態は表示と一致しているか

想定される状態

室温が 22-24 度の場合：

- シリアルモニタ：「状態: 通常・消灯」
- LED：消えている

室温が 25 度以上の場合：

- シリアルモニタ：「状態: やや高温・ゆっくり点滅」
- LED：1 秒ごとに点滅

トラブルシューティング

問題：LED が期待通りに動かない

- 現在の温度を確認（シリアルモニタ）
- その温度で正しい動作をしているか確認
- LED の配線を確認（D13 に接続されているか）
- プログラムの条件を確認（>= の向きなど）

問題：シリアルモニタに何も表示されない

- ボーレートが 9600 になっているか確認
- 正しい COM ポートか確認
- USB ケーブルの接続を確認

§ 8. ステップ 5：実験

実験 1：手で温める

実験の進め方

1. 「センサーを手で包んでください」
2. 「シリアルモニタと LED を観察してください」
3. 「温度が上がる様子を見てください」
4. 「25 度を超えたら何が起きますか？」

5. 「さらに温めて、28 度を超えたら？」

観察のポイント

- 温度の変化：徐々に上昇する
- 25 度超過：LED が点滅し始める（ゆっくり）
- 28 度超過：点滅が速くなる
- シリアルモニタの表示も変わる

学生への質問

- 「今、何度ですか？」
- 「LED はどうなっていますか？」
- 「なぜそうなりましたか？」

💡 **ポイント：**条件と動作の対応を意識させる。「if 文のおかげで、温度によって自動的に動作が変わっている」ことを強調。

実験 2：冷ます

実験の進め方

1. 「手を離してください」
2. 「温度が下がる様子を観察してください」
3. 「28 度を下回ったら何が起きますか？」
4. 「25 度を下回ったら？」

観察のポイント

- 温度の変化：徐々に下降する
- 28 度未満：点滅がゆっくりになる
- 25 度未満：LED が消える

学びの強調：「上がるときも下がるときも、自動的に判断して動作が変わりますね。これが IoT システムの基本です」

実験 3：閾値の変

閾値（しきいち）の説明

「閾値」という言葉を説明する。

- 動作が変わる境界の温度のこと
- 今は 28 度と 25 度が閾値
- この値を変えると、システムの動作が変わる

実験内容

変更 1：28 を 30 に変更

```
if (temp >= 30) {  // 28 → 30
```

- 書き込んで実験
- 30 度まで温めないと速く点滅しない
- より高温にならないと反応しない

変更 2：25 を 23 に変更

```
else if (temp >= 23) {  // 25 → 23
```

- 書き込んで実験
- 室温でもゆっくり点滅することがある

- より低温でも反応する

学生への質問

- 「閾値を変えると、何が変わったか？」
- 「実際のシステムでは、どうやって閾値を決めると思うか？」

実用例の紹介

- エアコン：設定温度が閾値
- 冷蔵庫：食品の保存温度が閾値
- 温室：作物に適した温度が閾値

💡 **時間調整のポイント**

- 時間が余ったら：複数の閾値を試させる
- 時間が足りなかったら：実験 3 は宿題にする
- 早く終わった学生：発展課題に取り組ませる

§ 9. まとめ

今日の振り返り

学んだことの確認

以下の項目を確認しながら振り返る：

- **if 文の基本**：条件によって処理を分岐できる
- **比較演算子**：>=、<、==などを使って条件を書ける
- **else if と else**：複数の条件を順番に判定できる
- **順序の重要性**：厳しい条件を先に書く
- **入力と出力の統合**：センサー値に応じて動作を変えられる

IoT システムの基本構造

板書またはスライドで示す：

入力（センサー）
　　↓
判断（if 文）
　　↓
出力（LED）

強調ポイント：

「これが IoT システムの基本。エアコンも冷蔵庫も自動運転も、全部この仕組みで動いている。その基礎を身につけた」

次回の予告

Part4 の内容

- 複数のセンサーを組み合わせる
- データを SD カードに保存する
- LCD ディスプレイに表示する
- より実用的なシステムを作る

準備してもらうこと

- 今日の回路はそのままにしておく
- if 文の復習をしておく
- ワークブックを提出

§ 10. トラブルシューティング

プログラム関連のトラブル

問題 1：LED が点滅しない

原因候補：

- 温度が閾値に達していない
- LED の配線ミス
- LEDPIN の定義が間違っている
- pinMode の設定忘れ

確認手順：

1. シリアルモニタで現在の温度を確認
2. その温度で期待される動作を確認
3. LED が D13 に接続されているか確認
4. プログラムの LEDPIN 13 を確認
5. setup()に pinMode(LEDPIN, OUTPUT); があるか確認

問題 2：コンパイルエラー

頻出エラー：

エラー	原因	対処法
expected ';'	セミコロン忘れ	指摘された行の前の行にセミコロンを追加
expected '}'	括弧の閉じ忘れ	{ と } の数を数えて、不足分を追加
'elseif' was not declared	else if を elseif と書いている	else と if の間にスペースを入れる
ISO C++ forbids comparison	== を = と書いている	比較は == を使う

問題 3：期待と違う動作

シナリオ：30 度なのにゆっくり点滅する

原因：条件の順序が間違っている

```
// 間違った順序
if (temp >= 25) {
    ゆっくり点滅
```

```
} else if (temp >= 28) {
    速く点滅  // この条件は実行されない！
}
```

解決策：順序を入れ替える

```
// 正しい順序
if (temp >= 28) {
    速く点滅
} else if (temp >= 25) {
    ゆっくり点滅
}
```

ハードウェア関連のトラブル

問題 4：センサーが nan を表示

Part2 と同じ対処法：

- 数秒待つ（初回は失敗することがある）
- 配線を確認（特に VCC と DATA）
- センサーの向きを確認
- それでもダメならセンサー交換

理解度に関するトラブル

学生が「分からない」と言ったら

if 文の概念が理解できない場合：

- 日常生活の例で説明し直す
- フローチャートを一緒に描く
- 具体的な数字で動作を追ってみる

比較演算子が分からない場合：

- 数直線を描いて視覚的に説明
- 「以上」と「より大きい」の違いを強調
- クイズ形式で繰り返し練習

順序の重要性が分からない場合：

- 実際に順序を間違えたコードを見せる
- デバッガで動作を追ってみる
- フローチャートで流れを確認

§ 11. 発展課題（進度が早い学生向け）

チャレンジ 1：4 段階制御

課題内容

3 段階ではなく、4 段階に分けて LED を制御する

- 30 度以上：非常に速く点滅（100ms）
- 28 度以上 30 度未満：速く点滅（200ms）
- 25 度以上 28 度未満：ゆっくり点滅（1000ms）

- 25 度未満：消灯

学習ポイント

- else if を 3 つ使う練習
- 条件の順序をより意識する
- 細かい制御ができることを体験

ヒント

```
if (temp >= 30) {
    // 100ms で点滅
} else if (temp >= 28) {
    // 200ms で点滅
} else if (temp >= 25) {
    // 1000ms で点滅
} else {
    // 消灯
}
```

チャレンジ 2：湿度も使う

課題内容

温度と湿度の両方を条件に使う

- 温度 28 度以上 かつ 湿度 70%以上：非常に速く点滅
- 温度 28 度以上：速く点滅
- 温度 25 度以上：ゆっくり点滅
- それ以外：消灯

学習ポイント

- 論理演算子 && の使い方
- 複数条件の組み合わせ
- 実用的な不快指数判定に近い

ヒント

```
if (temp >= 28 && humi >= 70) {
    // 暑くてジメジメ
    digitalWrite(LEDPIN, HIGH);
    delay(100);
    digitalWrite(LEDPIN, LOW);
    delay(100);
} else if (temp >= 28) {
    // 暑いけどまだマシ
    digitalWrite(LEDPIN, HIGH);
    delay(200);
    digitalWrite(LEDPIN, LOW);
    delay(200);
} else if (temp >= 25) {
```

```
    // ゆっくり点滅
    ...
}
```

チャレンジ 3：不快指数の計算

課題内容

温度と湿度から不快指数を計算して、それに応じて制御する

不快指数の計算式

```
float discomfort = 0.81 * temp + 0.01 * humi * (0.99 * temp - 14.3) + 46.3;
```

不快指数の目安

- 85 以上：暑くてたまらない → 非常に速く点滅
- 80-85：暑くて汗が出る → 速く点滅
- 75-80：やや暑い → ゆっくり点滅
- 75 未満：快適 → 消灯

プログラム例

```
float temp = dht.readTemperature();
float humi = dht.readHumidity();

// 不快指数を計算
float discomfort = 0.81 * temp + 0.01 * humi * (0.99 * temp - 14.3) + 46.3;

Serial.print("不快指数: ");
Serial.println(discomfort);

if (discomfort >= 85) {
    // 非常に速く点滅
} else if (discomfort >= 80) {
    // 速く点滅
} else if (discomfort >= 75) {
    // ゆっくり点滅
} else {
    // 消灯
}
```

学習ポイント

- float 型の計算
- 複数データの組み合わせ
- 実用的な指標の活用

§ 12. よくある質問への回答

if 文について

Q1: if 文の括弧は必ず必要ですか？

A: はい、丸括弧()と中括弧{}の両方が必要です。ただし、処理が1行だけの場合は中括弧を省略できますが、初心者には推奨しません。

```
// 推奨（初心者向け）
if (temp >= 25) {
    digitalWrite(13, HIGH);
}

// 省略形（上級者向け、非推奨）
if (temp >= 25) digitalWrite(13, HIGH);
```

Q2: 条件を複数書けますか？

A: はい、&& や || を使えば複数の条件を組み合わせられます。

- && : かつ（両方とも正しいとき）
- || : または（どちらか正しいとき）

```
// 両方の条件が正しいとき
if (temp >= 28 && humi >= 70) {
    Serial.println("暑くてジメジメ");
}

// どちらかが正しいとき
if (temp >= 30 || humi >= 80) {
    Serial.println("注意");
}
```

比較演算子について

Q3: == と = の違いは何ですか？

A: これは非常に重要な違いです。

- ==（イコール2つ）: 比較。「等しいかどうか」を調べる
- =（イコール1つ）: 代入。「値を入れる」という意味

```
// 比較（if文で使う）
if (temp == 25) {  // 温度が25度かどうか
    ...
}

// 代入（値を入れる）
temp = 25;  // tempに25を入れる
```

よくある間違い：

```
// 間違い！
if (temp = 25) {  // これは代入になってしまう
    ...
}
```

```
// 正しい
if (temp == 25) {  // 比較
    ...
}
```

Q4: >= と > の違いは？

A: 境界値を含むか含まないかの違いです。

- >=：以上。その値も含む
- >：より大きい。その値は含まない

```
// temp = 25 のとき
temp >= 25  // true（25は25以上）
temp > 25   // false（25は25より大きくない）

// temp = 26 のとき
temp >= 25  // true
temp > 25   // true
```

else if と else について

Q5: else if の順序を変えても大丈夫ですか？

A: いいえ、順序は非常に重要です。

if文は上から順番にチェックし、最初に当てはまった条件だけを実行します。

```
// 正しい順序（厳しい条件が先）
if (temp >= 28) {
    速く点滅
} else if (temp >= 25) {
    ゆっくり点滅
}
// temp = 30 のとき → 速く点滅

// 間違った順序
if (temp >= 25) {
    ゆっくり点滅
} else if (temp >= 28) {
    速く点滅  // この条件は実行されない！
}
// temp = 30 のとき → ゆっくり点滅（間違い）
```

Q6: else は必ず必要ですか？

A: いいえ、必須ではありません。

```
// elseなし（どの条件にも当てはまらない場合、何もしない）
if (temp >= 28) {
    速く点滅
} else if (temp >= 25) {
```

```
    ゆっくり点滅
}
// temp = 20 のとき → 何もしない

// else あり（明示的に何もしないことを書く）
if (temp >= 28) {
    速く点滅
} else if (temp >= 25) {
    ゆっくり点滅
} else {
    digitalWrite(LEDPIN, LOW);  // 消灯
}
// temp = 20 のとき → 消灯
```

今回は「消灯」も明示的に書くので、else を使います。

delay について

Q7: delay()を使うと、温度測定も遅れませんか？

A: その通りです。delay()の間は、Arduino は他のことができません。

しかし、今回の場合は問題ありません。なぜなら：

- 温度はゆっくりしか変わらない
- 数秒の遅れは許容範囲
- DHT11 自体が測定に 2 秒かかる

より高速な処理が必要な場合：

millis()という関数を使います。これは時間を計りながら、他の処理も同時にできます。

```
// millis()を使った例（発展）
unsigned long previousMillis = 0;
const long interval = 1000;  // 1 秒

void loop() {
    unsigned long currentMillis = millis();

    // 1 秒経過したか？
    if (currentMillis - previousMillis >= interval) {
        previousMillis = currentMillis;
        // LED の点滅処理
    }

    // この間も他の処理ができる
}
```

これは少し難しいので、今日は delay()で十分です。

閾値について

Q8: 閾値はどうやって決めるのですか？

A: システムの目的によって決めます。

実用例：

- **エアコン制御：**快適温度は 22-26 度なので、その範囲を基準に
- **農業用温室：**作物によって最適温度が違う（トマト 25-30 度、イチゴ 15-20 度）
- **食品保管：**食品安全のため、10 度以下を保つ
- **サーバールーム：**機器の故障を防ぐため、28 度以下を保つ

決め方のステップ：

1. 目的を明確にする（何を守りたいか、何を最適化したいか）
2. 基準となる値を調べる（規格、ガイドライン、過去のデータ）
3. 実験して調整する（実際に動かして最適値を見つける）
4. 余裕を持たせる（ギリギリではなく、少し余裕を持った値に）

§ 13. 授業改善のヒント

授業後の振り返り

チェック項目

- 時間配分は適切だったか？
- 学生の理解度は十分だったか？
- つまずきが多かった箇所はどこか？
- デモンストレーションは効果的だったか？
- 板書は分かりやすかったか？
- 発展課題に取り組めた学生はどれくらいか？

改善のポイント

- 時間が足りなかった → どの部分を削るか、前もって準備するか
- 時間が余った → 発展課題を増やす、実験時間を延ばす
- 理解度が低い → 説明方法を変える、例を増やす
- エラーが多い → チェックポイントを増やす、サンプルコードを用意

次回への引き継ぎ事項

記録しておくこと

- 理解度が低かった学生のリスト
- 特に優秀だった学生のリスト
- 頻出したエラーや質問
- 効果的だった説明方法
- 改善が必要な箇所

Part4 への準備

- 今日の回路はそのまま使う
- SD カードモジュールの準備
- LCD ディスプレイの準備
- if 文の復習が必要な学生のフォロー

学生からのフィードバック活用

ワークブックから読み取る情報

- **感想欄**：面白かった点、難しかった点
- **理解度チェック**：自己評価の傾向
- **記入の丁寧さ**：真剣に取り組んだか
- **質問欄**：共通の疑問点

次回授業への反映

- 共通の質問 → 授業冒頭で全体に説明
- 難しかった点 → 説明方法を改善
- 面白かった点 → さらに発展させる
- 個別の疑問 → 個別フォロー

§ 14. 参考資料

完成プログラム（再掲）

```
#include <DHT.h>

#define DHTPIN 2
#define DHTTYPE DHT11
#define LEDPIN 13

DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  dht.begin();
  pinMode(LEDPIN, OUTPUT);
}

void loop() {
  float temp = dht.readTemperature();
  float humi = dht.readHumidity();

  Serial.print("温度: ");
  Serial.print(temp);
  Serial.println("℃");

  Serial.print("湿度: ");
  Serial.print(humi);
  Serial.println("%");

  if (temp >= 28) {
```

```
    Serial.println("状態: 高温・速く点滅");
    digitalWrite(LEDPIN, HIGH);
    delay(200);
    digitalWrite(LEDPIN, LOW);
    delay(200);
  } else if (temp >= 25) {
    Serial.println("状態: やや高温・ゆっくり点滅");
    digitalWrite(LEDPIN, HIGH);
    delay(1000);
    digitalWrite(LEDPIN, LOW);
    delay(1000);
  } else {
    Serial.println("状態: 通常・消灯");
    digitalWrite(LEDPIN, LOW);
  }

  delay(2000);
}
```

配線図

Part1 と Part2 の配線をそのまま使用します。

LED 回路（Part1 から）

- LED アノード（長い足）→ D13
- LED カソード（短い足）→ 抵抗（220Ω）→ GND

DHT11 センサー（Part2 から）

- VCC（1 番ピン）→ Arduino 5V
- DATA（2 番ピン）→ Arduino D2
- GND（3 番ピン）→ Arduino GND

関連リソース

Arduino 公式ドキュメント

- **if statement:** https://www.arduino.cc/reference/en/language/structure/control-structure/if/
- **Comparison Operators:** https://www.arduino.cc/reference/en/language/structure/comparison-operators/
- **Boolean Operators:** https://www.arduino.cc/reference/en/language/structure/boolean-operators/

DHT11 ライブラリ

- **Adafruit DHT Sensor Library:** https://github.com/adafruit/DHT-sensor-library

IoT 技術応用
実習１ デバイス制御
Part4: データ記録と表示 教員用指導ガイド

§ 1. 授業概要

授業の位置づけ

Part4 は、全 4 回の実習の集大成です。

- **Part1**：出力（LED 制御の基礎）
- **Part2**：入力（センサーでデータ取得）
- **Part3**：統合（条件による制御）
- **Part4**：完成（データ記録と表示）← 今回

今回の授業で、実用的な IoT システムの全機能が揃います。学生にとって達成感のある回となるよう、丁寧に指導しましょう。

学習目標

知識面の目標

- SD カードへのデータ記録の仕組みを理解する
- LCD ディスプレイへの文字表示方法を理解する
- SPI 通信と I2C 通信の違いを理解する
- 複数モジュールの統合方法を理解する
- IoT システムの実用化に必要な要素を理解する

技能面の目標

- SD カードモジュールを正しく配線・接続できる
- LCD ディスプレイを正しく配線・接続できる
- ファイルへのデータ書き込みプログラムを作成できる
- LCD 表示プログラムを作成できる
- 複数の機能を統合したシステムを構築できる

態度面の目標

- 4 回の実習を通じた成長を実感する
- 実用的なシステムを作れたことに達成感を持つ
- 今後の学習への意欲を高める

項目	内容
導入	振り返り、目標説明、デモ
ステップ 1：SD カードモジュール準備	説明、配線

ステップ 2：SD カード配線	配線作業、確認
ステップ 3：SD カードプログラム	プログラム入力、検証
ステップ 4：動作確認	書き込み、SD カード確認
ステップ 5：LCD 準備・配線	説明、配線、ライブラリ
ステップ 6：LCD プログラム	プログラム追加
ステップ 7：最終動作確認	総合確認
まとめ	振り返り、感想、今後

必要な機材

学生 1 人あたり

- Arduino Uno（Part3 の回路付き）
- SD カードモジュール
- microSD カード（32GB 以下、FAT32 フォーマット済み）
- LCD ディスプレイ（16×2、I2C 接続）
- ジャンパーワイヤー（オス-オス）×10 本
- USB-A to B ケーブル
- PC（Arduino IDE インストール済み）
- ワークブック

教員用

- デモ用 Arduino（完成品）
- プロジェクター/大型ディスプレイ
- ホワイトボード/黒板
- 予備の SD カードモジュール（5 個程度）
- 予備の LCD ディスプレイ（5 個程度）
- 予備の SD カード（複数枚）
- SD カードリーダー（複数個）

💡 **重要：**SD カードは事前に FAT32 でフォーマットしておくこと。

§ 2. 授業準備

事前準備チェックリスト

- ワークブックの印刷・製本
- SD カードの購入・フォーマット
- LCD ディスプレイの動作確認

- デモ用プログラムの動作確認
- 予備部品の在庫確認
- すべての SD カードを FAT32 でフォーマット
- LCD のアドレス確認（0x27 or 0x3F or 0x50）
- サンプルプログラムの準備（USB メモリ等）
- SD カードリーダーの準備

SD カードの事前準備

フォーマット手順（Windows）

1. SD カードをパソコンに挿入
2. エクスプローラーで SD カードを右クリック
3. 「フォーマット」を選択
4. ファイルシステム：「FAT32」を選択
5. 「開始」をクリック

フォーマット手順（Mac）

1. SD カードを Mac に挿入
2. 「ディスクユーティリティ」を開く
3. SD カードを選択
4. 「消去」をクリック
5. フォーマット：「MS-DOS（FAT）」を選択
6. 「消去」をクリック

⚠ 重要な注意点

- 64GB 以上の SD カードは FAT32 でフォーマットできない場合があります
- 32GB 以下の SD カードを使用してください
- 古い SD カードでも問題ありません（512MB、1GB など）
- SDHC カードが最適です

LCD ディスプレイの事前確認

アドレスの確認

LCD の I2C アドレスは製品によって異なります。事前に確認しておきましょう。

一般的なアドレス：

- 0x27（最も一般的）
- 0x3F/0x50（一部の製品）

確認方法：

I2C スキャナーのスケッチを使用するか、製品の仕様書を確認してください。

バックライトの確認

- 裏側の可変抵抗でコントラストを調整できます
- 事前に見やすい状態に調整しておくとスムーズです

デモ用システムの準備

授業の冒頭で実際に動作するシステムを見せることが重要です。

デモの内容

1. LCD に温度・湿度が表示されている様子
2. 2 秒ごとに数字が更新される様子
3. LED が温度に応じて点滅する様子
4. SD カードを取り出してパソコンで開く
5. data.txt の中身を見せる
6. Excel でグラフ化した例を見せる

デモのポイント

- LCD をカメラで拡大表示
- シリアルモニタも大きく表示
- パソコンでのファイル確認をスムーズに
- グラフ例を事前に作成しておく

板書計画

授業中に板書する内容を計画しておきましょう。

左側：SD カード（SPI）

SD カード（SPI 通信）

GND → GND

VCC → 5V

MISO → D12（固定）

MOSI → D11（固定）

SCK → D13（固定）

CS → D10（変更可）

中央：LCD（I2C）

LCD（I2C 通信）

GND → GND

VCC → 5V

SDA → A4（固定）

SCL → A5（固定）

右側：SPI と I2C の比較

SPI：高速、ピン多い

I2C：低速、ピン少ない

複数デバイス接続可

§ 3. 導入

これまでの振り返り

話す内容

- Part1-3 で何を学んできたか
- それぞれの回で追加された機能
- 今日で完成形になること

学生への質問例

- 「Part1 では何をしましたか？」→「LED を制御した」

- 「Part2 では？」→「温度と湿度を測定した」
- 「Part3 では？」→「条件によって動作を変えた」
- 「今日は何をすると思いますか？」→「データを記録・表示」

強調ポイント：

「今日で、本格的な IoT システムが完成します。実際の現場で使われているシステムと同じ構造です」

今日の目標説明とデモ

デモンストレーション手順

1. **LCD の表示**
    1. 「これが今日の完成形です」
    2. LCD に温度・湿度が表示されている
    3. 「パソコンなしで、その場で情報が見られます」
    4. 「2 秒ごとに更新されます」
2. **SD カードの記録**
    1. 「そして、この SD カードにデータが記録されています」
    2. 電源を切って SD カードを取り出す
    3. パソコンに挿入して data.txt を開く
    4. 「このように、データが保存されています」
    5. 「後から分析できます」
3. **グラフ化の例**
    1. 事前に作成した Excel グラフを表示
    2. 「このようにグラフ化もできます」
    3. 「温度の変化が一目で分かりますね」

実用例の紹介

- **農業**：温室の環境管理（24 時間記録）
- **食品**：倉庫の温度管理（法律で記録義務）
- **博物館**：美術品の環境保護
- **研究**：実験データの自動記録

⚠️ **導入での注意点**

- デモが失敗しないよう、事前に十分テストする
- SD カードの取り出しをスムーズに（練習しておく）
- パソコンでの表示をプロジェクターで大きく
- 学生が「面白そう」「やってみたい」と感じられる雰囲気作り
- 最終回なので、達成感を持てるような前向きな言葉かけ

§ 4. ステップ 1：SD カードモジュールの準備

SD カードモジュールの説明

教え方のポイント

実物を見せながら説明：

1. SD カードモジュールの実物を見せる
2. 「これで、普通の SD カードを読み書きできます」
3. 「デジカメやスマホで使うのと同じ SD カードです」
4. 6 本のピンを指差しながら説明

SPI 通信の簡単な説明：

- 「SPI という通信方式を使います」
- 「高速ですが、ピンが多いです」
- 「MISO、MOSI、SCK は決まったピンに繋ぎます」
- 「CS だけは好きなピンに繋げます（今回は D10）」

板書：

ピンの対応を板書して、全員で確認します。

SD カードの準備

実演しながら説明：

1. **「SD カードは 32GB 以下のものを使います」**
2. **「今日使う SD カードは、FAT32 でフォーマット済みです」**
3. **「そのまま使えます」**
4. **SD カードを持って実演**
5. **「裏表があります。金色の端子がある方を下に」**
6. **「カチッと音がするまで押し込みます」**
7. **「それでは、SD カードをモジュールに差し込んでください」**

巡回確認：

全員がしっかり差し込んでいるか確認します。

配線の説明

配線表の提示：

プロジェクターまたは板書で配線表を表示します。

SD カード	Arduino	備考
GND	GND	黒いワイヤー推奨
VCC	5V	赤いワイヤー推奨
MISO	D12	固定（SPI で決まっている）
MOSI	D11	固定（SPI で決まっている）
SCK	D13	固定（SPI で決まっている）

CS	D10	変更可能（今回は D10）

重要ポイントの強調：

- **「D13 は LED と SD カードの両方に繋がります」**
- **「これは問題ありません。両方とも使えます」**
- **「データを書き込むとき、LED が少し点滅することがあります」**

💡 **よくある質問への回答**

Q: D13 が LED と SD カードの両方に繋がっていて大丈夫ですか？

A: 大丈夫です。D13 は SPI の SCK ピンとして使われますが、LED 制御にも同時に使えます。データ書き込み中に LED が少し点滅することがありますが、システムの動作には影響ありません。

Q: CS ピンは D10 以外でも良いですか？

A: はい、D10 以外のピンでも構いません。ただし、プログラムで指定するピン番号と実際の配線を一致させる必要があります。今回は全員 D10 で統一します。

Q: なぜ MISO、MOSI、SCK は固定なのですか？

A: これらは Arduino のハードウェア SPI 通信で使用する専用ピンです。Arduino Uno では、SPI を使う場合は必ずこれらのピンを使います。

§ 5. ステップ 2：SD カード配線

配線前の確認

安全確認：

- 「配線を始める前に、USB ケーブルを抜いてください」
- 「全員抜きましたか？隣の人と確認し合ってください」
- 「Part3 の回路はそのままで大丈夫です」

配線手順の指導

一つずつ確認しながら進める

ステップ 1：GND 接続

1. 「まず、GND を繋ぎます」
2. 「SD カードモジュールの GND から、黒いワイヤーで Arduino の GND へ」
3. デモ機で実演
4. 「繋ぎましたか？」
5. 30 秒ほど待つ

ステップ 2：VCC 接続

1. 「次に、VCC を繋ぎます」
2. 「SD カードモジュールの VCC から、赤いワイヤーで Arduino の 5V へ」
3. 「3.3V ではありません。5V です」
4. デモ機で実演
5. 30 秒ほど待つ

ステップ 3-5：SPI ピン接続

1. 「次は、MISO、MOSI、SCK を繋ぎます」

2. 「MISO は D12 へ」（実演）
3. 「MOSI は D11 へ」（実演）
4. 「SCK は D13 へ。LED と一緒ですが問題ありません」（実演）
5. 「一つずつ確認しながら繋いでください」
6. 1 分ほど待つ

ステップ 6：CS 接続

1. 「最後に、CS を D10 へ繋ぎます」（実演）
2. 「これで配線完了です！」
3. 30 秒ほど待つ

指導のコツ

- 一つのピンを繋ぐたびに、全員ができるまで待つ
- 焦らせず、一つずつ確実に
- デモ機を回して見せるか、プロジェクターで拡大表示
- 間違えそうなポイント（VCC が 3.3V でなく 5V）を繰り返し強調

配線確認と巡回

全体確認

「それでは、配線を確認しましょう。以下の項目をチェックしてください」

- GND は GND に繋がっているか
- VCC は 5V に繋がっているか（3.3V ではない）
- MISO、MOSI、SCK は正しいピンか
- CS は D10 か
- SD カードはしっかり差し込まれているか

「確認できたら、隣の人と見せ合ってください」

巡回チェックポイント

- VCC が 5V に繋がっているか（最重要）
- MISO/MOSI/SCK のピン番号
- SD カードの差込み（カチッと音がしたか）
- 配線の抜けや緩み

よくある間違い：

- VCC を 3.3V に繋いでいる → 5V に修正
- MISO/MOSI が逆 → 正しく接続し直す
- SD カードが浅く差し込まれている → しっかり押し込む

⚠️ 配線トラブルの予防

この段階で配線ミスを見つけることが重要です。後でプログラムを書き込んでから気づくと、時間がかかります。

- 全員の配線を目視で確認する
- 特に VCC の接続を重点的にチェック
- 不安な学生には個別に確認

§ 6. ステップ 3：SD カード書き込みプログラム

プログラムの全体像説明

説明内容：

- 「今からプログラムを入力します」
- 「Part3 のプログラムに、SD カード機能を追加します」
- 「追加する内容は 3 つです」
- 「1. SD ライブラリの読み込み」
- 「2. SD カードの初期化」
- 「3. ファイルへの書き込み」
- 「プログラムは少し長いですが、焦らず正確に入力してください」

プログラム入力

入力の進め方

1. 投影しながら説明：

- プロジェクターでプログラムを投影
- ブロックごとに区切って入力させる
- 各ブロックの説明を簡単に

2. 入力のペース配分：

セクション	時間	ポイント
ライブラリと define	2 分	SD と SPI の追加、CSPIN
setup 関数	4 分	SD.begin()の理解
loop 関数（前半）	3 分	Part3 と同じ
ファイル書き込み	4 分	File、open、close の理解
if 文と残り	2 分	Part3 と同じ

説明のポイント

SD カード初期化：

```
if (!SD.begin(CSPIN)) {
  Serial.println("SD カード初期化失敗");
  while (1);
}
```

- 「!は否定です。初期化が失敗したら、という意味」
- 「while(1)で、そこで停止します」
- 「SD カードが認識できないと、先に進みません」

ファイルへの書き込み：

```
File dataFile = SD.open("data.txt", FILE_WRITE);
if (dataFile) {
  dataFile.print("温度:");
  dataFile.print(temp);
  dataFile.close();
}
```

- 「SD.open で、data.txt を開きます」
- 「FILE_WRITE は、書き込みモードです」
- 「dataFile.print()で、ファイルに書き込みます」
- 「Serial.print()と同じ使い方です」
- 「最後に close()で、ファイルを閉じます」
- 「close を忘れると、データが保存されません」

プログラム確認

検証手順：

1. 「入力が終わったら、チェックマークをクリックしてください」
2. 「検証が始まります」
3. 「エラーが出た人は手を挙げてください」
4. 巡回してエラーを確認

よくあるエラーと対処：

エラーメッセージ	原因	対処法
expected ';'	セミコロン忘れ	エラー行を確認して追加
expected '}'	括弧の対応ミス	括弧の数を確認
'SD' was not declared	#include 忘れ	#include <SD.h>を追加
'File' does not name a type	#include <SD.h>忘れ	ライブラリを追加

§ 7. ステップ 4：動作確認

書き込みと初期確認

書き込み手順

1. 「それでは、プログラムを書き込みましょう」
2. 「矢印ボタンをクリックしてください」
3. 「書き込みには 30 秒ほどかかります」
4. 待機
5. 「書き込みが完了したら、シリアルモニタを開いてください」

シリアルモニタの確認

表示される内容：

SD カード初期化中...

SD カード初期化成功

温度：23.5℃

湿度：45.2%

データ保存成功

確認ポイント：

- 「SD カード初期化成功」と表示されているか
- 温度と湿度が表示されているか
- 「データ保存成功」と表示されているか
- 2 秒ごとに更新されるか

トラブルシューティング

症状	原因	対処
SD カード初期化失敗	配線ミス、SD カード不良	配線確認、SD カード差し直し
ファイルオープン失敗	SD カード容量不足	SD カードをフォーマット
何も表示されない	シリアルモニタの設定	ボーレート 9600 を確認

SD カード内のファイル確認

確認手順の説明

1. 「それでは、SD カードを取り出して確認しましょう」
2. 「まず、USB ケーブルを抜いてください」
3. 「SD カードを取り出します。爪で押すと出てきます」
4. デモ機で実演
5. 「パソコンに差し込んでください」
6. 「SD カードの中を開いて、data.txt を探してください」
7. 1 分ほど待つ
8. 「見つかりましたか？」
9. 「ダブルクリックして開いてください」

ファイルの内容確認

プロジェクターで教員の PC の画面を表示し、data.txt を開いて見せる

温度:23.5,湿度:45.2

温度:23.6,湿度:45.3

温度:23.5,湿度:45.1

「このように、温度と湿度が記録されていますね！」

「2 秒ごとに 1 行ずつ追加されています」

戻す作業

1. 「確認できたら、SD カードを Arduino に戻してください」
2. 「カチッと音がするまで押し込んでください」
3. 「USB ケーブルを繋いでください」
4. 「また動き始めましたね」

💡 **SD カードリーダーについて**

学生の PC に SD カードスロットがない場合：

- USB SD カードリーダーを各机に配布
- または、数台を共用して順番に確認
- 時間がかかる場合は、教員の PC で代表して確認

§ 8. ステップ 5：LCD ディスプレイの準備

LCD ディスプレイの説明

実物を見せながら説明

1. 「次は、LCD ディスプレイを使います」
2. LCD を持って見せる
3. 「16 文字×2 行の文字を表示できます」
4. 「パソコンなしで、その場で情報を確認できます」

I2C 通信の説明

- 「I2C という通信方式を使います」
- 「SD カードの SPI とは違う方式です」
- 「I2C は、ピンが 4 本だけで済みます」
- 「GND、VCC、SDA、SCL の 4 本です」
- 「SDA と SCL は、Arduino で決まったピンに繋ぎます」
- 「Arduino Uno では SDA は A4、SCL は A5 です」

板書：SPI と I2C の比較

	SPI（SD カード）	I2C（LCD）
ピン数	6 本	4 本
速度	高速	低速
用途	SD カード、高速センサー	LCD、RTC、各種センサー

LCD 配線

配線手順

配線前の確認：

「USB ケーブルを抜いてください」

ステップごとの指導：

1. 「LCD の GND を、Arduino の GND に繋ぎます」（実演）
2. 30 秒待つ
3. 「LCD の VCC を、Arduino の 5V に繋ぎます」（実演）
4. 30 秒待つ
5. 「LCD の SDA を、Arduino の A4 に繋ぎます」（実演）
6. 「A4 はアナログピンです。D4 ではありません」（強調）
7. 30 秒待つ
8. 「LCD の SCL を、Arduino の A5 に繋ぎます」（実演）
9. 「A5 もアナログピンです。D5 ではありません」（強調）
10. 30 秒待つ
11. 「配線完了です！簡単でしたね」

よくある間違い

- **SDA を D4 に繋ぐ：**「A4 です！デジタルピンではありません」
- **SCL を D5 に繋ぐ：**「A5 です！デジタルピンではありません」
- **VCC を 3.3V に繋ぐ：**「5V に繋いでください」

配線確認と巡回

確認項目

- GND は GND に繋がっているか
- VCC は 5V に繋がっているか
- SDA は A4 に繋がっているか（D4 ではない）
- SCL は A5 に繋がっているか（D5 ではない）

「確認できたら、隣の人と見せ合ってください」

巡回チェック

特に SDA/SCL の接続を重点的に確認します。A4/A5 でなく D4/D5 に繋いでいる学生が必ずいます。

⚠ A4/A5 と D4/D5 の混同に注意

これは非常によくある間違いです。

- Arduino Uno では、アナログピンは A0〜A5 で表示されています
- デジタルピンは D0〜D13 です
- 混同しやすいので、何度も強調しましょう
- 巡回時に必ず確認してください

§ 9. ステップ 6：LCD 表示プログラム

ライブラリのインストール

手順の説明

1. 「LCD を使うには、ライブラリが必要です」
2. 「Arduino IDE で、『スケッチ』→『ライブラリをインクルード』→『ライブラリを管理』を選んでください」
3. スクリーンで実演
4. 「検索欄に『LiquidCrystal I2C』と入力してください」

5. 「『LiquidCrystal I2C by Frank de Brabander』が見つかります」
6. 「『インストール』をクリックしてください」
7. 1 分ほど待つ
8. 「インストールが完了したら、閉じてください」

トラブル対応

- ライブラリが見つからない → スペルを確認
- インストールできない → インターネット接続を確認
- 別のライブラリが表示される → 作者名を確認（Frank de Brabander）

プログラムへの追加

追加する内容の説明

「先ほどのプログラムに、LCD 表示機能を追加します」

「追加する場所は 4 箇所です」

1. ライブラリの追加：

```
#include <LiquidCrystal_I2C.h>
```

「#include <SPI.h> の下に 1 行追加してください」

2. LCD オブジェクトの作成：

```
LiquidCrystal_I2C lcd(0x27, 16, 2);
```

「DHT dht の下に 1 行追加してください」

「0x27 は LCD のアドレスです」

「16 は文字数、2 は行数です」

3. setup 関数に LCD 初期化を追加：

```
lcd.init();
lcd.backlight();
lcd.print("Starting...");
```

「Serial.println("SD カード初期化成功"); の下に 3 行追加してください」

4. loop 関数に LCD 表示を追加：

```
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("Temp: ");
lcd.print(temp);
lcd.print("C");
lcd.setCursor(0, 1);
lcd.print("Humi: ");
lcd.print(humi);
lcd.print("%");
```

「Serial.println("%"); の下に 9 行追加してください」

コマンドの説明

コマンド	意味

lcd.init();	LCD を初期化する
lcd.backlight();	バックライトを点灯する
lcd.clear();	画面を消去する
lcd.setCursor(列, 行);	カーソル位置を設定する
lcd.print("文字");	文字を表示する

setCursor の説明：

- 「setCursor(0, 0)は、1 行目の左端です」
- 「setCursor(0, 1)は、2 行目の左端です」
- 「列は 0〜15、行は 0〜1 です」

プログラム確認

「それでは、チェックマークをクリックして検証してください」

よくあるエラー

- 'LiquidCrystal_I2C' does not name a type → ライブラリ未インストール
- expected ';' → セミコロン忘れ
- スペルミス（LiquidCrystal のスペル）

§ 10. ステップ 7：最終動作確認

総合確認

書き込みと確認

1. 「それでは、完成したプログラムを書き込みましょう！」
2. 「矢印ボタンをクリックしてください」
3. 30 秒ほど待つ
4. 「書き込みが完了したら、LCD を見てください」
5. 「まず『Starting...』と表示されます」
6. 「そして、温度と湿度が表示されます！」

全機能の確認

「今、すべての機能が同時に動いています！」

- LCD に温度・湿度が表示されているか
- 2 秒ごとに更新されるか
- シリアルモニタにも表示されているか
- 「データ保存成功」と表示されているか
- LED が温度に応じて点滅しているか

感動の瞬間：

「これで、完全な IoT システムの完成です！」

「おめでとうございます！」

拍手を促す

トラブルシューティング

症状	原因	対処法
LCD に何も表示されない	配線ミス、電源不足	配線確認、VCC が 5V か確認
文字化けする	アドレスが違う	0x27 を 0x3F に変更
バックライトだけ点灯	コントラスト調整	裏の可変抵抗を回す
表示がおかしい	SDA/SCL が逆	A4 と A5 を確認

LCD アドレスの変更方法

0x27 で動かない場合：

```
LiquidCrystal_I2C lcd(0x3F, 16, 2);
```

「0x27 を 0x3F か 0x50 に変更して、もう一度書き込んでください」

§ 11. まとめ

今日の成果確認

話す内容

今日作ったシステムの確認：

- 「今日作ったシステムには、4 つの機能があります」
- 「1. 温度と湿度を測定する」
- 「2. 温度によって LED の動作を変える」
- 「3. データを SD カードに保存する」
- 「4. LCD ディスプレイに表示する」
- 「これは、本格的な IoT システムです」

4 回の実習の振り返り：

回	内容	学んだこと
Part1	LED 制御	出力の基本
Part2	センサー	入力の基本
Part3	条件分岐	入出力の統合
Part4	記録・表示	システムの完成

「これらは、すべての IoT システムの基本です」

データの活用方法

実用的な活用例：

1. **Excel でグラフ化：**
   - data.txt を Excel で開く
   - カンマで区切る
   - 折れ線グラフを作成
   - 温度の変化を視覚化
2. **長期間の記録：**
   - 1 週間、1 ヶ月とデータを記録
   - 季節による変化を分析
   - 異常値の検出
3. **分析と改善：**
   - データから問題を発見
   - 改善策を検討
   - 効果を数値で確認

「皆さんが作ったシステムは、実際の現場で使われているものと同じです」

発展的な学習

今後の学習の提案：

レベル 1：センサーを増やす

- 気圧センサー、照度センサーなど

レベル 2：Wi-Fi 通信

- ESP32 を使ってクラウドに送信
- スマホから確認

レベル 3：自動制御

- 扇風機やヒーターを制御
- 完全自動のシステム

「興味がある人は、ぜひ挑戦してみてください！」

§ 12. 片付けと提出

片付け手順

SD カードについて

- 「SD カードは返却してください」
- 「data.txt をコピーしたい人は、USB メモリに保存してください」
- 「コピーしたい人は、手を挙げてください」
- コピーのサポート

回路について

- 「今日の回路は、このままでも大丈夫です」

- 「または、部品を外して元に戻してください」
- 「ワイヤーやセンサーは、ケースに戻してください」

ワークブックの提出

- 「ワークブックを提出してください」
- 「記入漏れがないか確認してください」
- 「特に、考察問題と感想を書いてください」

回収物の確認

- SD カードモジュール
- SD カード
- LCD ディスプレイ
- ジャンパーワイヤー
- ワークブック

確認事項：

- 破損した部品はないか
- 紛失した部品はないか
- 全員のワークブックを回収したか

§ 13. トラブルシューティング総合

SD カード関連のトラブル

問題 1：SD カード初期化失敗

症状：「SD カード初期化失敗」と表示される

原因：

- SD カードが差し込まれていない
- 配線ミス（特に CS ピン）
- SD カードのフォーマットが違う
- SD カードが破損している

対処法：

1. SD カードを差し直す
2. 配線を確認（特に CS→D10）
3. 別の SD カードで試す
4. SD カードを FAT32 で再フォーマット

問題 2：data.txt が作成されない

症状：SD カードにファイルがない

原因：

- ファイルオープンに失敗している
- 書き込み前に電源を切った

対処法：

1. シリアルモニタで「データ保存成功」を確認
2. しばらく動作させてから電源を切る
3. SD カードの容量を確認

問題 3：データが追記されない

症状：最初の 1 行だけで更新されない

原因：

- FILE_WRITE が正しく指定されていない
- close()が呼ばれていない

対処法：

1. SD.open("data.txt", FILE_WRITE)を確認
2. dataFile.close()があるか確認

LCD 関連のトラブル

問題 4：LCD に何も表示されない

症状：バックライトも点灯しない

原因：

- 配線ミス（特に VCC）
- A4/A5 でなく D4/D5 に接続

対処法：

1. VCC が 5V に繋がっているか確認
2. SDA が A4、SCL が A5 か確認（D4/D5 ではない）
3. GND が正しく接続されているか確認

問題 5：バックライトだけ点灯

症状：光るが文字が見えない

原因：

- コントラストの調整が必要
- I2C アドレスが違う

対処法：

1. LCD の裏側の青い可変抵抗を回す
2. 0x27 を 0x3F に変更して再書き込み

問題 6：文字化けする

症状：意味不明な文字が表示される

原因：

- I2C アドレスが違う
- SDA/SCL の配線ミス

対処法：

1. 0x27 を 0x3F に変更
2. SDA と SCL の配線を再確認

プログラム関連のトラブル

問題 7：コンパイルエラー

よくあるエラーメッセージと対処：

エラーメッセージ	原因	対処法

'SD' was not declared	#include <SD.h>忘れ	ライブラリを追加
'LiquidCrystal_I2C' does not name a type	ライブラリ未インストール	ライブラリをインストール
expected ';' before	セミコロン忘れ	前の行にセミコロン追加
expected '}' at end of input	括弧の対応ミス	括弧の数を確認

問題 8：動作が不安定

症状：時々動作が止まる、表示がおかしくなる

原因：

- 電源不足
- 配線の接触不良
- 同時に多くの処理

対処法：

1. USB ポートを変える（電源容量の大きいポート）
2. 配線をしっかり差し直す
3. lcd.clear()の頻度を減らす

§ 14. 発展課題の指導

発展課題の位置づけ

発展課題は、早く終わった学生や興味のある学生向けです。

- 無理に全員に実施させる必要はない
- 時間に余裕があれば紹介
- 自宅学習の課題としても活用可能

チャレンジ 1：タイムスタンプの追加

内容

データに記録時刻を追加する

実装のヒント

millis()関数を使った簡易版：

```
unsigned long elapsed = millis() / 1000;  // 秒に変換
dataFile.print(elapsed);
dataFile.print(",温度:");
dataFile.print(temp);
```

RTC モジュールを使った完全版：

DS3231 などの RTC モジュールを追加することで、正確な日時を記録できます。

指導のポイント

- millis()は起動からの経過時間であることを説明
- 本格的にはRTCモジュールが必要であることを伝える
- 時間があれば、RTCモジュールのデモを見せる

チャレンジ2：データのグラフ化

内容

SDカードのdata.txtをExcelで開いてグラフを作成する

手順

1. data.txtをExcelで開く
2. 「データ」→「区切り文字」でカンマを選択
3. データを列に分ける
4. 温度の列を選択して「挿入」→「グラフ」
5. 折れ線グラフを選択

分析の視点

- 最高値・最低値を見つける
- 平均値を計算する
- 温度と湿度の関係を見る
- 時間経過による変化を観察

指導のポイント

- Excelが使えない学生にはサポート
- Google スプレッドシートでも同様に可能
- グラフ作成の基本を教える良い機会

チャレンジ3：LCD表示のカスタマイズ

内容

LCD表示を工夫して、より見やすく、情報豊かにする

アイデア例

- 温度が高いときに「WARNING!」と表示
- 快適度を「Comfortable」「Hot」などで表示
- アイコンや記号を使う（°、%など）
- 2行をスクロールさせて情報を切り替える

サンプルコード

```
// 温度による表示切替
if (temp >= 28) {
  lcd.setCursor(0, 1);
  lcd.print("*** HOT! ***");
} else if (temp >= 25) {
  lcd.setCursor(0, 1);
  lcd.print("** WARM **");
} else {
  lcd.setCursor(0, 1);
  lcd.print("Comfortable");
```

```
}
```

チャレンジ 4：アラーム機能の追加

内容

温度が閾値を超えたら、ブザーを鳴らす

必要な部品

- 圧電ブザー
- ジャンパーワイヤー×2 本

配線

- ブザーの+を D8 に接続
- ブザーの-を GND に接続

プログラム例

```
#define BUZZERPIN 8

void setup() {
  pinMode(BUZZERPIN, OUTPUT);
}

void loop() {
  if (temp >= 30) {
    tone(BUZZERPIN, 1000, 100);  // 1000Hz、100ms
  }
}
```

注意点

- ブザーは音が大きいので注意
- tone()関数の使い方を説明
- 周波数や音の長さを変えて実験できる

§ 15. よくある質問への回答

システム全般について

Q1：なぜ SD カードと LCD の両方が必要なのですか？

A：それぞれ役割が違います。

- **SD カード**：長期間のデータ保存、後からの分析
- **LCD**：リアルタイムでの確認、パソコンなしで使用

実用システムでは両方あると便利です。

Q2：このシステムは何日間動かし続けられますか？

A：理論上は無制限ですが、実際には：

- SD カードの容量が許す限り
- 電源が供給される限り
- 部品が故障しない限り

1GB の SD カードでも数ヶ月分のデータを保存できます。

Q3: 複数のセンサーを同時に使えますか？

A: はい、可能です。

- アナログピンは 6 個（A0-A5）
- デジタルピンは 14 個（D0-D13）
- I2C は同じピン（A4,A5）で複数デバイス接続可能

ピン数の範囲内で、多くのセンサーを追加できます。

SD カードについて

Q4: なぜ 32GB 以下なのですか？

A: FAT32 フォーマットの制約です。

- SD ライブラリは FAT32 のみ対応
- 64GB 以上は exFAT でフォーマットされることが多い
- 32GB 以下なら確実に FAT32 を使用できる

Q5: データはどこまで増えますか？

A: 計算してみましょう。

- 1 行あたり約 30 バイト
- 2 秒ごとに 1 行 = 1 日で 43,200 行
- 43,200 行 × 30 バイト ≒ 1.3MB/日
- 1GB の SD カードで約 2 年分

実際には十分な容量があります。

Q6: data.txt を削除したい場合は？

A: パソコンで削除できます。

1. SD カードをパソコンに挿入
2. data.txt を削除
3. SD カードを Arduino に戻す
4. 新しい data.txt が自動作成される

LCD について

Q7: なぜアドレスが 0x27 と 0x3F の 2 種類あるのですか？

A: メーカーによって設定が違います。

- I2C アドレスは製造時に設定される
- 多くは 0x27 だが、一部は 0x3F
- どちらも正しく、プログラムで指定する必要がある

Q8: LCD に日本語を表示できますか？

A: この LCD では日本語表示は困難です。

- 16×2 キャラクタ LCD は英数字専用
- 日本語対応の LCD モジュールもある（高価）
- グラフィック LCD なら表示可能（複雑）

Q9: LCD の表示が遅いです。速くできますか？

A: lcd.clear()を減らすと改善します。

```
// 毎回 clear しない方法
```

```
lcd.setCursor(6, 0);
lcd.print("    ");  // スペースで消す
lcd.setCursor(6, 0);
lcd.print(temp);
```

clear()は画面全体を消すため時間がかかります。

通信方式について

Q10: SPI と I2C はどう使い分けますか？

A: 用途によって選択します。

項目	SPI	I2C
速度	高速（数 MHz）	低速（100-400kHz）
配線	多い（4 本+CS）	少ない（2 本）
複数デバイス	CS 毎に 1 ピン必要	同じピンで複数接続
用途	高速通信が必要 （SD カード、センサー）	低速で OK、ピン節約 （LCD、RTC、センサー）

Q11: 両方同時に使えますか？

A: はい、今回のように同時使用できます。

- SPI と I2C は独立した通信方式
- 使用するピンが異なる
- 干渉せずに同時動作可能

§ 17. 授業改善のヒント

授業後の振り返り

チェック項目

- 時間配分は適切だったか？
- SD カードの配線で混乱はなかったか？
- LCD 配線での A4/A5 と D4/D5 の混同は防げたか？
- プログラム入力に十分な時間が取れたか？
- 動作確認でトラブルはなかったか？
- 学生は達成感を感じられたか？
- 発展課題に取り組めた学生はいたか？

改善のポイント

- **時間が足りなかった：**
    - プログラムのファイル配布を検討
    - 配線説明を簡略化

    - 実験時間を短縮
- **時間が余った：**
    - 発展課題の実施
    - データ分析の時間を増やす
    - 4回の総まとめディスカッション
- **トラブルが多かった：**
    - 配線図をより分かりやすく
    - チェックポイントを増やす
    - TA（ティーチングアシスタント）の配置

学生からのフィードバック活用

ワークブックから読み取る情報

- **感想欄：**
    - 達成感を感じられたか
    - 難しかった点
    - 面白かった点
    - もっとやりたいことはあるか
- **理解度チェック：**
    - どの部分の理解が不足しているか
    - 全体的な習熟度
- **考察問題：**
    - 応用力があるか
    - システム全体を理解しているか

次への改善

- 共通の質問・疑問 → 説明を強化
- 難しかった点 → 教材や説明方法を改善
- 面白かった点 → さらに発展させる
- 要望 → 次年度のカリキュラムに反映

4回シリーズ全体の評価

全体を通した成果

Part4終了後、4回シリーズ全体を評価しましょう。

評価項目

- カリキュラムの難易度は適切だったか
- 時間配分は適切だったか
- 段階的な学習ができたか
- 最終的なシステムは実用的か
- 学生の満足度は高いか

次年度への引き継ぎ事項

- 効果的だった教授方法
- 改善が必要な箇所
- 部品の推奨品

- よくあるトラブルと対処法
- 発展課題のアイデア

§ 18. 参考資料

完成プログラム全文

```
#include <DHT.h>
#include <SD.h>
#include <SPI.h>
#include <LiquidCrystal_I2C.h>

#define DHTPIN 2
#define DHTTYPE DHT11
#define LEDPIN 13
#define CSPIN 10

DHT dht(DHTPIN, DHTTYPE);
LiquidCrystal_I2C lcd(0x27, 16, 2);

void setup() {
  Serial.begin(9600);
  dht.begin();
  pinMode(LEDPIN, OUTPUT);

  Serial.println("SD カード初期化中...");
  if (!SD.begin(CSPIN)) {
    Serial.println("SD カード初期化失敗");
    while (1);
  }
  Serial.println("SD カード初期化成功");

  lcd.init();
  lcd.backlight();
  lcd.print("Starting...");
}

void loop() {
  float temp = dht.readTemperature();
  float humi = dht.readHumidity();

  Serial.print("温度: ");
```

```
Serial.print(temp);
Serial.println("℃");
Serial.print("湿度: ");
Serial.print(humi);
Serial.println("%");

lcd.clear();
lcd.setCursor(0, 0);
lcd.print("Temp: ");
lcd.print(temp);
lcd.print("C");
lcd.setCursor(0, 1);
lcd.print("Humi: ");
lcd.print(humi);
lcd.print("%");

File dataFile = SD.open("data.txt", FILE_WRITE);
if (dataFile) {
  dataFile.print("温度:");
  dataFile.print(temp);
  dataFile.print(",湿度:");
  dataFile.println(humi);
  dataFile.close();
  Serial.println("データ保存成功");
} else {
  Serial.println("ファイルオープン失敗");
}

if (temp >= 28) {
  digitalWrite(LEDPIN, HIGH);
  delay(200);
  digitalWrite(LEDPIN, LOW);
  delay(200);
} else if (temp >= 25) {
  digitalWrite(LEDPIN, HIGH);
  delay(1000);
  digitalWrite(LEDPIN, LOW);
  delay(1000);
} else {
  digitalWrite(LEDPIN, LOW);
```

```
  }

  delay(2000);
}
```

配線図まとめ

完成システムの全配線

デバイス	ピン	Arduino
LED	アノード（長い足）	D13
	カソード（短い足）	220Ω 抵抗 → GND
DHT11	VCC	5V
	DATA	D2
	GND	GND
SD カード	GND	GND
	VCC	5V
	MISO	D12
	MOSI	D11
	SCK	D13
	CS	D10
LCD	GND	GND
	VCC	5V
	SDA	A4
	SCL	A5

関連リソース

Arduino 公式ドキュメント

- SD Library: https://www.arduino.cc/en/Reference/SD
- SPI Library: https://www.arduino.cc/en/Reference/SPI
- LiquidCrystal Library: https://www.arduino.cc/en/Reference/LiquidCrystal

ライブラリ

- DHT Sensor Library (Adafruit): https://github.com/adafruit/DHT-sensor-library
- LiquidCrystal I2C: https://github.com/johnrickman/LiquidCrystal_I2C

IoT 技術応用
実習 2 クラウド接続
導入編 教員用指導ガイド

§ 🎯 指導目標

- 実習 2 の全体像と学習目標を理解させる
- ESP32 の特徴と実習 1 との違いを明確に説明する
- IoT システムの 4 層構造を理解させる
- 学生のモチベーションを高め、実習への期待感を醸成する

§ ⏱️ 説明ポイント

内容	ポイント
実習 2 の概要説明	実習 1 との違いを強調
ESP32 の特徴	Wi-Fi 内蔵の利点
IoT システムの構成	4 層構造を図解
実習の流れと注意点	安全面の注意喚起
質疑応答とまとめ	学生の理解度確認

§ 📝 指導の流れ

1. オープニング

🎤 **トーク例：**

「これから、実習 2『IoT 通信とクラウド接続』を始めます。実習 1 では、センサーでデータを取得しましたね。実習 2 では、そのデータをインターネットに送信します。つまり、**センサーとクラウドをつなぐ**のが今回の目標です。」

指導ポイント：

- 実習 1 の復習を簡潔に
- 実習 2 の新規性を明確に説明
- 「なぜクラウドに送る必要があるのか」を問いかける

⚠️ **よくある学生の疑問：**

- Q: 「シリアルモニタで見るだけではダメなんですか？」

- A: 「シリアルモニタは、パソコンと USB でつながっている時だけ見られます。クラウドに送れば、スマホからでも、外出先からでも見られます。これが IoT の強みです。」

2. ESP32 の特徴

説明すべき 3 つのポイント：

1. **Wi-Fi/Bluetooth 内蔵** - 追加モジュール不要
2. **高性能** - Arduino Uno の 15 倍の処理速度
3. **低価格** - 高性能なのに安い

📊 **視覚的な比較を推奨：**

Arduino Uno と ESP32 の写真やスペック表を並べて提示すると効果的です。特に以下の点を強調：

- 動作周波数: 16MHz → 240MHz（15 倍）
- RAM: 2KB → 520KB（260 倍）
- Wi-Fi: なし → あり（最大の違い）

3. IoT システムの 4 層構造

各層の説明：

層	役割	実習 2 での実装
センシング層	データ取得	DHT22 センサー
デバイス層	データ処理・通信	ESP32
ネットワーク層	データ伝送	Wi-Fi、HTTP
クラウド層	データ蓄積・可視化	Ambient/ThingSpeak

🎨 **板書・スライドのアイデア：**

4 層を縦に並べた図を描き、データが下から上に流れる様子を矢印で示すと効果的です。各層に「実習 2 で使うもの」を書き込むと、学生の理解が深まります。

4. 実習の流れと安全注意事項

4 つの Part の概要：

- **Part 1:** Wi-Fi 接続の基本（ESP32 のネットワーク機能）
- **Part 2:** センサーデータ取得（DHT22 の使い方）
- **Part 3:** クラウドへのデータ送信（核心部分）
- **Part 4:** データ分析（可視化と読み取り）

⚠️ **安全上の注意（必ず伝達）：**

- **配線の確認:** VCC と GND を絶対に逆に接続しないこと
- **通電中の配線変更禁止:** 必ず USB を抜いてから配線を変更
- **静電気対策:** 冬季は特に注意
- **個人情報:** Wi-Fi パスワードや API キーを他人に見せない

5. 質疑応答とまとめ

📌 **確認すべき理解度チェックポイント**

- 実習 2 の最終目標は何か？（センサーデータをクラウドに送信し可視化すること）
- ESP32 の最大の特徴は？（Wi-Fi 内蔵）
- IoT システムの 4 層とは？（センシング、デバイス、ネットワーク、クラウド）

§ 🔧 事前準備チェックリスト

【授業前日までに】

- Arduino IDE がインストールされているか確認
- ESP32 ボードサポートがインストール可能か確認
- Wi-Fi アクセスポイントの動作確認（2.4GHz 対応必須）
- Ambient または ThingSpeak のアカウント作成手順確認
- 教室のネットワーク環境確認（ファイアウォール設定）
- 機材の動作確認（ESP32、DHT22 センサー）
- 予備機材の準備（故障時用）

【授業当日】

- プロジェクターまたはモニターの動作確認
- サンプルコードの動作確認
- Wi-Fi の電波強度確認
- クラウドサービスへのアクセス確認
- ワークブックとナレーション資料の準備

§ 💡 指導のコツ

1. 実習 1 との対比を活用

実習 1 でできたこと・できなかったことを明確にし、実習 2 で何が可能になるかを示すと、学生の理解が深まります。

2. 「つながる」ことの価値を強調

IoT の本質は「つながること」です。センサーがインターネットにつながることで、どんな新しい価値が生まれるか、具体例を示しましょう。

- スマート農業：畑の状態を遠隔監視
- スマートホーム：外出先から家の環境確認
- 環境モニタリング：都市全体の気温・湿度マップ

3. 段階的な理解を促す

一度に全てを理解させようとせず、Part 1 から順に、段階的に理解を深めさせましょう。導入編では「全体像」を示すだけで十分です。

4. 失敗を学びの機会に

「うまくいかない」ことは当然あります。トラブルシューティングを通じて、より深い理解が得られることを伝えましょう。

§ ❓ よくある質問と回答例

Q1: ESP32 は Arduino Uno より難しいですか？

A: 基本的な使い方は Arduino Uno と同じです。ただし、Wi-Fi 通信など新しい機能が追加されるため、学ぶことは増えます。しかし、ライブラリが充実しているので、実は簡単に使えます。

Q2: 自宅の Wi-Fi でも実習できますか？

A: はい、できます。ただし、2.4GHz 帯の Wi-Fi が必要です。最近の 5GHz 専用ルーターでは接続できないので注意してください。

Q3: クラウドサービスは有料ですか？

A: Ambient と ThingSpeak は、どちらも無料プランがあります。実習で使う範囲なら、無料プランで十分です。

Q4: プログラミングが苦手でもできますか？

A: 大丈夫です。サンプルコードを提供しますし、ワークブックに詳しい説明があります。わからないところは、質問してください。

📊 **学生の理解度評価のポイント**

評価項目	確認方法
実習の目的理解	「実習 2 で何を作りますか？」と質問し、「センサーからクラウドまでの IoT システム」と答えられるか
ESP32 の特徴理解	「ESP32 の最大の特徴は？」と質問し、「Wi-Fi 内蔵」と答えられるか
4 層構造の理解	「IoT システムの 4 つの層は？」と質問し、4 層を挙げられるか
安全意識	「配線で最も注意すべき点は？」と質問し、「VCC と GND を逆にしない」と答えられるか

§ 🎯 導入編の成功基準

導入編が成功したと言えるのは、以下の状態です：

- ✅ 全学生が実習 2 の全体像を理解している
- ✅ ESP32 と Arduino Uno の違いを説明できる
- ✅ IoT システムの構成要素を理解している
- ✅ 安全上の注意点を認識している
- ✅ 実習への期待感とモチベーションが高まっている

IoT 技術応用
実習 2 クラウド接続
Part1: Wi-Fi 接続とネットワーク基礎 教員用指導ガイド

§ Part 1 の指導目標🎯

- ESP32 の Wi-Fi 機能を理解させる
- Wi-Fi 接続の仕組み（SSID、パスワード、IP アドレス）を説明する
- ESP32 の開発環境セットアップを完了させる
- Wi-Fi 接続プログラムを実装し、動作確認まで到達させる

§ 指導ポイント

内容	指導ポイント
ESP32 の特徴説明	Arduino Uno との比較
Wi-Fi 通信の仕組み	3 ステップを図解
開発環境セットアップ	ボードマネージャの使い方
Wi-Fi 接続プログラム	コードの各部分を丁寧に
動作確認	シリアルモニタで接続確認
トラブル対応とまとめ	よくあるエラーの解決

§ 詳細指導手順📝

1. ESP32 の特徴説明

Part 1 では、ESP32 を Wi-Fi に接続します。実習 1 で使った Arduino Uno には、Wi-Fi 機能がありませんでした。ESP32 には、Wi-Fi が**内蔵**されています。これが最大の違いです。

比較表を使った説明：

項目	Arduino Uno	ESP32
動作周波数	16 MHz	240 MHz（15 倍）
RAM	2 KB	520 KB（260 倍）
Wi-Fi	なし	**内蔵（2.4GHz）**
Bluetooth	なし	内蔵

💡 学生に考えさせる質問：

「なぜ IoT では、Wi-Fi 内蔵のマイコンが重要なのでしょうか？」

期待する回答例：

- 追加の部品が不要（コスト削減）
- 配線が簡単（開発が楽）
- 消費電力が少ない
- 小型化できる

2. Wi-Fi 通信の仕組み

3 ステップで説明：

ステップ 1：アクセスポイントの検索

- ESP32 が周囲の Wi-Fi を探す
- SSID（ネットワーク名）を検出
- スマホで Wi-Fi 一覧を見るのと同じ

ステップ 2：認証

- SSID とパスワードでログイン
- セキュリティチェック
- 許可されたデバイスのみ接続可能

ステップ 3：IP アドレスの取得

- DHCP サーバーから自動取得
- IP アドレス = ネットワーク上の住所
- これでインターネットに接続完了

📊 板書・スライドのアイデア：

3 つのステップを矢印でつなぎ、各ステップで何が起きているかを図解すると効果的です。特に、DHCP による IP アドレス自動取得は、学生にとって新しい概念なので、丁寧に説明しましょう。

3. 開発環境セットアップ

⚠️ 重要な注意点：

この作業は**初回のみ**必要です。すでに ESP32 サポートをインストール済みの場合はスキップしてください。

手順（実演しながら説明）：

1. **環境設定 URL の追加**
    - 「ファイル」→「環境設定」
    - 「追加のボードマネージャの URL」に以下を入力：

https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json

2. **ボードマネージャでインストール**
    - 「ツール」→「ボード」→「ボードマネージャ」
    - 検索欄に「esp32」と入力
    - 「esp32 by Espressif Systems」をインストール
    - ⏱️ 数分かかることを伝える
3. **ボード選択**
    - 「ツール」→「ボード」→「ESP32 Dev Module」

🔧 よくあるトラブルと対処法：

- **ダウンロードが遅い：** ネットワークの問題。時間がかかる場合があることを伝える
- **インストール失敗：** Arduino IDE を再起動して再試行
- **ボードが見つからない：** URL が正しく入力されているか確認

4. Wi-Fi 接続プログラムの説明

プログラムの構造（各部分を順に説明）：

① ライブラリのインクルード

```
#include <WiFi.h>
```

→ Wi-Fi 機能を使うための宣言

② 接続情報の設定

```
const char* ssid = "あなたの SSID";
const char* password = "あなたのパスワード";
```

→ **必ず自分の Wi-Fi 情報に変更**することを強調

③ setup 関数での接続開始

```
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
}
```

→ 接続完了まで待機するループ

④ IP アドレスの表示

```
Serial.println(WiFi.localIP());
```

→ 接続成功の確認

⑤ loop 関数での接続監視

```
if (WiFi.status() != WL_CONNECTED) {
    WiFi.reconnect();
}
```

→ 切断時の自動再接続

🎯 各部分で必ず説明すべきポイント：

- WiFi.begin()：接続開始の命令
- WiFi.status()：現在の接続状態を取得
- WL_CONNECTED：接続成功を示す定数
- WiFi.localIP()：割り当てられた IP アドレス
- WiFi.reconnect()：再接続の試行

5. 動作確認

実演手順：

1. **SSID とパスワードの変更**
    - 必ず自分の環境に合わせること
    - ダブルクォーテーションを忘れないこと
2. **ポートの選択確認**
    - 「ツール」→「シリアルポート」
    - 正しいポートが選択されているか
3. **書き込み実行**
    - 「書き込み」ボタンをクリック
    - 30 秒〜1 分程度かかることを伝える
4. **シリアルモニタで確認**
    - ボーレート：115200 に設定
    - 「Wi-Fi 接続成功」と IP アドレスが表示されれば OK

💡 成功例の画面を見せる：

事前に撮影したシリアルモニタの成功画面を見せると、学生が何を目指せばよいか明確になります。

6. トラブルシューティング

問題	解決方法
「接続できません」と表示	1. SSID とパスワードを再確認（大文字・小文字、全角・半

	角） 2. 2.4GHz 帯の Wi-Fi か確認（5GHz は非対応） 3. 電波強度を確認（アクセスポイントに近づく）
シリアルモニタが文字化け	ボーレートを 115200 に設定（プログラムとシリアルモニタの両方）
書き込みエラー	1. USB ケーブルの接続確認 2. 正しいポートが選択されているか 3. ESP32 のボード選択を確認
接続は成功するが不安定	1. アクセスポイントに近づく 2. 他の Wi-Fi 機器との干渉を確認 3. ループ内の delay 時間を調整

§ 事前準備チェックリスト🔧

- Wi-Fi アクセスポイントが 2.4GHz 対応か確認
- 実習環境場所の電波強度が十分か確認
- SSID とパスワードを学生に伝達（または各自で設定）
- ESP32 ボードサポートのインストール完了
- サンプルコードの動作確認完了
- 予備の ESP32 を準備

§ 指導のコツ💡

1. 段階的な理解を促す

Wi-Fi 接続は複雑なプロセスですが、「SSID 探す → ログイン → IP もらう」という 3 ステップで簡潔に説明すると理解しやすくなります。

2. 実演を多用する

口頭説明だけでなく、実際に画面を見せながら操作することで、学生の理解が深まります。

3. エラーを恐れない姿勢を育てる

「うまく接続できないのは当然」という前提で、トラブルシューティングの重要性を伝えましょう。

4. 成功体験を共有する

最初の学生が接続に成功したら、クラス全体に伝えて雰囲気を盛り上げましょう。

§ Part 1 の成功基準🎯

- ✅ 全学生の ESP32 が Wi-Fi に接続できた
- ✅ シリアルモニタで IP アドレスが確認できた
- ✅ Wi-Fi 接続の 3 ステップを説明できる

- [x] プログラムの各部分の役割を理解している

IoT 技術応用
実習 2 クラウド接続
Part 2 :センサーデータ取得とシリアル通信 教員用指導ガイド

§ 🎯 Part 2 の指導目標

- DHT22 センサーの特徴と DHT11 との違いを理解させる
- センサーライブラリのインストールと使用方法を習得させる
- 正しい配線方法を身につけさせる（特に安全面）
- エラー処理の重要性を理解させる
- 不快指数の計算を通じて、データの意味を考えさせる

§ 指導ポイント

内容	指導ポイント
DHT22 の特徴説明	DHT11 との比較表
ライブラリインストール	実演しながら説明
配線の説明	安全注意を徹底
プログラム説明	エラー処理を強調
動作確認	不快指数の意味
トラブル対応	配線チェック徹底

§ 📝 詳細指導手順

1. DHT22 の特徴説明

Part 2 では、Part 1 の Wi-Fi 接続に、センサーを追加します。使うのは DHT22 という温湿度センサーです。実習 1 で使った DHT11 の**上位モデル**です。

項目	DHT11	DHT22	改善点
温度範囲	0〜50° C	-40〜80° C	**幅広い**
温度精度	±2° C	±0.5° C	**4 倍正確**
湿度範囲	20〜90%	0〜100%	**全範囲**
湿度精度	±5%	±2%	**2.5 倍正確**
測定間隔	1 秒	2 秒	やや遅い

💡 学生に考えさせる質問：

「なぜ実用的な IoT システムでは、DHT22 のような高精度センサーが必要なのでしょうか？」

期待する回答例：

- 正確なデータで適切な制御ができる
- 誤差が小さいと異常検知がしやすい
- 広い温度範囲で屋外でも使える

2. ライブラリのインストール

手順（実演）：

1. 「スケッチ」→「ライブラリをインクルード」→「ライブラリを管理」
2. 検索欄に「DHT sensor library」と入力
3. 「DHT sensor library by Adafruit」を選択
4. 「インストール」をクリック
5. 依存ライブラリのインストールを求められたら「すべてインストール」

🔧 インストール中の説明：

「ライブラリとは、複雑な処理を簡単に使えるようにした**部品集**のようなものです。DHT22 を使うための処理が、すべてこのライブラリに入っています。」

3. 配線の説明

⚠️ 安全上の注意（必ず強調）：

- **VCC と GND を絶対に逆に接続しない** → センサーが壊れます
- 配線変更時は必ず USB ケーブルを抜く
- ジャンパー線をしっかり挿す（接触不良防止）

DHT22 ピン	ESP32 ピン	役割
VCC（左端）	3.3V	電源供給
DATA（左から 2 番目）	GPIO 4	データ通信
NC（左から 3 番目）	未接続	使用しない
GND（右端）	GND	グラウンド

📷 配線図を見せる：

実物またはスライドで配線図を示し、特に **VCC と GND の色分け**（赤と黒など）を強調すると、学生が間違えにくくなります。

配線チェックポイント：

1. 「VCC は 3.3V に接続」 ✓
2. 「DATA は GPIO 4 に接続」 ✓
3. 「GND は GND に接続」 ✓
4. 「VCC と GND が逆になっていない」 ✓

4. プログラムの説明

重要な部分を順に説明：

① センサーの初期化

```
#include <DHT.h>
#define DHTPIN 4
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);

void setup() {
    dht.begin();  // センサー初期化
}
```

→ センサーを使う準備

② データの読み取り

```
float humidity = dht.readHumidity();
float temperature = dht.readTemperature();
```

→ 湿度と温度を取得

③ エラーチェック（最重要）

```
if (isnan(humidity) || isnan(temperature)) {
```

```
    Serial.println("センサー読み取りエラー");
    return;
}
```

→ **エラー処理は必須**

④ 不快指数の計算

```
float discomfortIndex = 0.81 * temperature + 0.01 * humidity *
                        (0.99 * temperature - 14.3) + 46.3;
```

→ 温度と湿度から快適さを数値化

🎯 エラー処理の重要性を強調：

「センサーは必ずしも正しいデータを返すとは限りません。配線不良、ノイズ、センサーの一時的な不調など、様々な原因でエラーが起きます。**isnan 関数でエラーチェック**をすることで、異常なデータをプログラムに取り込まないようにします。」

isnan = is Not a Number（数値でない）

5. 不快指数の説明

不快指数	体感
～55	寒い
55～60	肌寒い
60～65	何も感じない
65～70	快適
70～75	暑くない
75～80	やや暑い
80～85	暑くて汗が出る
85～	暑くてたまらない

💡 実践的な活用例：

「気象庁も、この不快指数を使って『蒸し暑さ』を予報しています。温度だけでなく、**湿度も快適さに影響する**ことがわかりますね。」

6. トラブルシューティング

問題	解決方法
「センサー読み取りエラー」連続	1. 配線を確認（特に VCC/GND 逆接続） 2. ジャンパー線の挿し直し 3. センサーを別のものに交換
温度が異常に高い	1. ESP32 の発熱の影響（センサーを離す） 2. 直射日光が当たっていないか 3. 配線ミス（VCC/GND が逆）
湿度が 0%または 100%	1. センサーの初期化失敗（再起動） 2. ライブラリの再インストール 3. センサーの故障
データが更新されない	1. delay(2000)が適切か確認 2. loop 関数が正しく動作しているか 3. シリアルモニタの自動スクロール確認

§ 🔧 事前準備チェックリスト

- DHT22 センサーの動作確認完了
- DHT ライブラリのインストール手順確認
- 配線図の準備（印刷またはスライド）
- 予備の DHT22 センサーを用意
- ジャンパー線の予備を用意
- サンプルコードの動作確認

§ 💡 指導のコツ

1. 配線時の安全を最優先

VCC/GND の逆接続でセンサーが壊れるケースが多いです。配線前に必ず全員で確認させましょう

2. エラー処理の重要性を体験させる

意図的にセンサーを抜いてエラーが発生する様子を見せると、エラー処理の必要性が実感できます

3. データの意味を考えさせる

温度・湿度の数値だけでなく不快指数という「意味のある指標」に変換することでデータ活用の重要性が理解できます

4. 実物を使った説明

DHT22 センサーを手に持って、各ピンを指し示しながら説明すると効果的です

§ 🎯 Part 2 の成功基準

- [x] 全学生が正しく配線できた（VCC/GND 逆接続ゼロ）
- [x] 温度・湿度・不快指数が正常に表示されている
- [x] エラー処理の重要性を理解している
- [x] 不快指数の意味を説明できる

IoT 技術応用
実習 2 クラウド接続
Part 3 ：クラウドサービス連携 教員用指導ガイド

§ Part 3 の指導目標🎯

- クラウド IoT サービスの意義と機能を理解させる
- Ambient または ThingSpeak のアカウント作成とチャンネル設定を完了させる
- API キーの概念とセキュリティの重要性を理解させる
- HTTP 通信によるデータ送信を実装させる
- **データがクラウドに届く瞬間の感動を体験させる**（最重要）

§ 指導ポイント

内容	指導ポイント
クラウド IoT サービス説明	「つながる」価値を強調
Ambient と ThingSpeak 紹介	どちらを選ぶかアドバイス
アカウント作成	実演しながら誘導
チャンネル設定	API キーの取得と保管
データ送信プログラム	ambient.send()の仕組み
送信確認と感動体験	グラフが動く瞬間を共有
トラブルシューティング	ネットワーク問題への対処

§ 詳細指導手順📝

1. クラウド IoT サービスの説明

「Part 3 は、実習 2 の**クライマックス**です。今まで教室の中だけで動いていたシステムが、インターネットを通じて**世界とつながります**。皆さんのスマホからでも、家に帰ってからでも、センサーのデータが見られるようになります。」

クラウド IoT サービスの 5 つの機能：

1. **データ受信** – デバイスからのデータを受け取る
2. **データ保存** – 時系列データとして蓄積
3. **可視化** – グラフやダッシュボードで表示
4. **API 提供** – 外部アプリからアクセス可能
5. **アラート** – 異常値の検知と通知

💡 「なぜクラウドが必要か」を問いかける：

質問:「シリアルモニタで見るのと、クラウドに送るのと、何が違いますか？」
期待する回答：

- USB ケーブルがなくても見られる
- 遠隔地から確認できる
- 複数人で同時に見られる
- 過去のデータを保存・分析できる
- スマホからも見られる

2. Ambient と ThingSpeak の比較

項目	Ambient	ThingSpeak
言語	日本語	英語
運営	日本企業	MathWorks（米国）
無料プラン	8 チャンネル 3000 データ/日	4 チャンネル 8200 メッセージ/日
設定難易度	★☆☆（簡単）	★★☆（やや複雑）
高度な分析	△	○（MATLAB 連携）
おすすめ対象	初心者	発展学習希望者

📌 どちらを選ぶかのアドバイス：

- **Ambient 推奨:** 初めて IoT を学ぶ学生、日本語で学びたい学生
- **ThingSpeak 推奨:** 英語に抵抗がない学生、MATLAB に興味がある学生、発展的な分析をしたい学生

※ クラス全体で統一する必要はありません。学生の興味に応じて選択させましょう。

3. アカウント作成

Ambient の場合：

1. https://ambidata.io/ にアクセス
2. 「ユーザー登録（無料）」をクリック
3. メールアドレスとパスワードを入力
4. 確認メールのリンクをクリックして認証
5. ログイン完了

ThingSpeak の場合：

1. https://thingspeak.com/ にアクセス
2. 「Sign Up」をクリック
3. MathWorks アカウントを作成（メールアドレス必須）
4. 確認メールのリンクをクリック
5. ログイン完了

⚠️ 注意点（必ず伝える）：

- 学校のメールアドレスが使えない場合、個人のメールアドレスを使用
- パスワードは忘れないようにメモ（安全な場所に）
- 確認メールが届かない場合は迷惑メールフォルダを確認

4. チャンネル設定と API キー取得

Ambient の場合：

1. ログイン後、「チャンネルを作る」をクリック
2. チャンネル名を入力（例：「教室の温湿度」）
3. 「作成」をクリック
4. **チャンネル ID** と**ライトキー**が表示される
5. この 2 つを**必ずメモ**する

ThingSpeak の場合：

1. 「New Channel」をクリック
2. Name 欄にチャンネル名を入力
3. Field 1 を「Temperature」、Field 2 を「Humidity」に設定
4. 「Save Channel」をクリック
5. 「API Keys」タブから **Write API Key** を確認
6. この **API キーを必ずメモ**する

🔐 API キーの重要性を説明：

「API キーは、**あなたのチャンネルにアクセスするための鍵**です。この鍵があれば、誰でもデータを送信できます。だから、**他人に見せてはいけません**。パスワードと同じように、大切に管理してくださ

い。」

セキュリティの基本: API キーは、GitHub などの公開リポジトリにアップロードしない！

5. データ送信プログラムの説明

Ambient ライブラリのインストール：

1. ライブラリマネージャを開く
2. 「Ambient ESP32 ESP8266 lib」を検索
3. インストール

プログラムの重要部分：

```
#include <Ambient.h>

WiFiClient client;
Ambient ambient;

// チャンネル情報（自分の情報に変更）
unsigned int channelId = 12345;  // ★必ず変更
const char* writeKey = "あなたのライトキー";  // ★必ず変更

void setup() {
    // Wi-Fi 接続後
    ambient.begin(channelId, writeKey, &client);
}

void loop() {
    float temperature = dht.readTemperature();
    float humidity = dht.readHumidity();

    // データを設定
    ambient.set(1, temperature);  // データ 1: 温度
    ambient.set(2, humidity);     // データ 2: 湿度

    // 送信
    if (ambient.send()) {
        Serial.println("送信成功");
    } else {
```

```
        Serial.println("送信失敗");
    }

    delay(30000);  // 30秒待機（Ambientの推奨間隔）
}
```

🎯 必ず説明すべきポイント：

- ambient.set(1, temperature) - データ1に温度を設定
- ambient.send() - クラウドに送信（この瞬間がIoT！）
- delay(30000) - 30秒間隔（頻繁すぎる送信を防ぐ）
- 送信の成功/失敗を確認する重要性

6. 送信確認と感動体験

✨ 「IoTの瞬間」を演出：

手順:

1. プログラムを書き込む
2. シリアルモニタで「送信成功」を確認
3. クラウドサービスのWebページを開く
4. **グラフが表示され、リアルタイムで更新される様子を確認**
5. スマートフォンからもアクセスしてみる

指導のコツ:

- 最初の1人が成功したら、クラス全体に伝えて雰囲気を盛り上げる
- 「今、皆さんのセンサーがインターネットにつながりました！」と強調
- グラフが更新される瞬間を一緒に見て、感動を共有する
- 「これがIoTです」と明確に伝える

7. トラブルシューティング

問題	解決方法
「送信失敗」が続く	1. チャンネルIDとライトキーが正しいか確認 2. Wi-Fi接続が安定しているか確認 3. スペースや改行が混入していないか確認
グラフにデータが表示されない	1. Webページを更新（F5キー） 2. 送信間隔が適切か確認（30秒以上） 3. シリアルモニタで「送信成功」が出ているか確認
HTTPエラー（403, 404等）	1. APIキーの入力ミス 2. チャンネルIDの入力ミス

	3. クラウドサービスのメンテナンス中
学校のネットワークで送信できない	1. ファイアウォール設定を確認 2. ネットワーク管理者に相談 3. 別のネットワーク（スマホのテザリング等）で試す

§ 事前準備チェックリスト🔧

- Ambient/ThingSpeak のデモアカウント準備
- 教室のネットワークでクラウドサービスにアクセス可能か確認
- サンプルコードの動作確認（実際に送信してグラフ表示確認）
- 学生用のメールアドレス準備またはメールアドレス持参の指示
- API キー管理の注意事項資料準備

§ 指導のコツ💡

1. 「つながる瞬間」を最大限に演出

グラフが初めて表示される瞬間は、学生にとって大きな感動体験です。この瞬間を逃さず、「これが IoT だ！」と強調しましょう。

2. API キーの重要性を実感させる

セキュリティの概念を教える絶好の機会です。API キーが漏れると何が起きるか、具体例を示すと効果的です。

3. ネットワークトラブルへの備え

学校のネットワーク環境によっては、送信できない場合があります。事前に確認し、代替案（スマホのテザリング等）を用意しましょう。

4. 成功例を共有

早く成功した学生のグラフを全員で見ることで、モチベーションが高まります。

§ Part３の成功基準🎯

- ✅ 全学生がクラウドサービスにアカウントを作成できた
- ✅ チャンネルが正しく設定され、API キーを取得できた
- ✅ センサーデータがクラウドに送信され、グラフで確認できた
- ✅ API キーの重要性とセキュリティを理解している
- ✅ **「データがつながる感動」を体験できた**（最重要）

IoT 技術応用
実習 2 クラウド接続
Part４：データ可視化と分析 教員用指導ガイド

§ Part４の指導目標🎯

- データ可視化の意義と目的を理解させる
- 時系列データの読み方（トレンド、周期性、変動、異常値）を習得させる
- グラフから意味を読み取る力を養う
- データ分析の 4 ステップ（観察→疑問→仮説→検証）を体験させる
- 実習 2 全体の達成感を共有し、今後の学習につなげる

§ 指導ポイント

内容	指導ポイント
データ可視化の意義	なぜグラフが重要か
時系列データの読み方	4 つの観点を説明
実際のグラフ分析	学生のデータを使用
データ分析の実践	観察→疑問→仮説→検証
システム改善の提案	発展的思考を促す
実習 2 全体の振り返り	達成感の共有

§ 詳細指導手順

1. データ可視化の意義

「Part 4 は、実習 2 の総仕上げです。Part 3 でクラウドにデータを送りました。そのデータ、ただ**見る**だけでいいでしょうか？いいえ、データは**読む**もの、**考える**ものです。そこから意味を見つけ出すのが、データ分析です。」

データ可視化の４つの目的：

目的	説明	具体例
理解しやすくする	数字の羅列→直感的に理解	1000 行の数値 vs 1 枚のグラフ
傾向を見つける	時間変化やパターンの発見	気温の日変化、週変化
異常を発見する	通常と異なる値を検出	突然の温度上昇
意思決定を支援する	データに基づく判断	エアコンをいつ ON にするか

💡 学生に考えさせる質問：

問: 「100 個の数字が並んでいるのと、グラフで見るのと、どちらが理解しやすいですか？」
期待する回答: グラフの方が理解しやすい → では、なぜ？ → 視覚的に見えるから

2. 時系列データの読み方

４つの観点を順に説明：

① トレンド（傾向）
- 全体として増加・減少・横ばいのどれか
- 例：午前中は気温が上昇、午後から下降
- **板書例:** 右上がりの矢印で「増加トレンド」を示す

② 周期性（パターン）
- 規則的に繰り返す変化
- 例：毎日 14 時頃に最高気温、朝方に最低気温
- **板書例:** 波線で「周期的変化」を示す

③ 変動の大きさ
- 値が大きく変わるか、安定しているか
- 例：室内は変動小、屋外は変動大
- **板書例:** 幅の異なる波で「変動の大きさ」を示す

④ 異常値
- 明らかに他と異なる値
- 例：突然 50 度になる → センサーエラーか実際の異常か
- **板書例:** グラフ上の突出点を丸で囲む

📊 実際のグラフを見せながら説明：

事前に準備した 24 時間のグラフを投影し、上記 4 つの観点を実際に指し示しながら説明すると、学生の理解が深まります。

3. 実際のグラフ分析

✨ 学生自身のデータを分析させる：

手順:

1. 各自のクラウドサービスページを開く
2. 24 時間のグラフを表示
3. 4 つの観点でグラフを観察
4. 気づいたことをワークブックに記入
5. 数名に発表してもらう

学生が気づくべきポイント（誘導質問）：

- **Q1:** 「温度が最も高いのは何時頃ですか？」
  → A: 午後 2 時頃（太陽の高度が最大の少し後）
- **Q2:** 「温度が最も低いのは何時頃ですか？」
  → A: 朝方（夜間の冷却後）
- **Q3:** 「温度が上がると、湿度はどうなりますか？」
  → A: 多くの場合下がる（空気の保水能力が上がるため）
- **Q4:** 「教室のエアコンをつけた時、グラフに変化が見られますか？」
  → A: 急激な温度変化として表れる可能性

💡 発見を共有させる：

学生が見つけた面白い発見を全体で共有させましょう。「私のグラフでは〜が見られました」という発表は、他の学生の分析意欲を刺激します。

4. データ分析の 4 ステップ実践

データ分析の思考プロセス：

ステップ 1: 観察
→ グラフから事実を読み取る
例: 「午後 2 時に温度が 28 度まで上昇している」
ステップ 2: 疑問
→ なぜそうなっているのか問いかける
例: 「なぜ午後 2 時に最高温度なのか？」
ステップ 3: 仮説

→ 原因を推測する

例：「太陽の高度が最大だから」「窓からの日射が強いから」

ステップ 4：検証

→ 仮説を確かめる方法を考える

例：「晴れの日と曇りの日で比較する」「窓にカーテンをして測定する」

🎯 グループワーク（推奨）：

2〜3 人のグループで、自分たちのグラフについて上記 4 ステップで議論させると、より深い学びになります。

時間： 5 分程度

発表： 各グループから 1 つずつ発見を共有

5. システム改善の提案

💡 「次にどう発展させるか」を考えさせる：

実習 2 で作ったシステムは、**基本システム**です。ここから、どう改善・発展させるか考えることが、エンジニアリング思考です。

3 つの方向での改善案：

方向	改善案例
ハードウェア	• CO2 センサーを追加 • 照度センサーで明るさも測定 • 複数のセンサーで部屋の温度分布を測定
ソフトウェア	• 温度が閾値を超えたらメール通知 • データの平均値・最大値・最小値を計算 • グラフに予測線を追加
データ分析	• 機械学習で異常検知 • 快適度指数の計算と可視化 • 他の教室との比較分析

🚀 学生のアイデアを引き出す質問：

- 「このシステムを、どこで使えると思いますか？」
- 「どんな機能があったら、もっと便利になりますか？」
- 「他にどんなセンサーを追加したいですか？」

6. 実習 2 全体の振り返り

✨ 達成感の共有（最重要）：

「Part 1 から Part 4 まで、たくさんのことを学びました。振り返ってみましょう。」

実習 2 で達成したこと：

1. **Part 1:** ESP32 を Wi-Fi に接続
2. **Part 2:** DHT22 センサーでデータ取得
3. **Part 3:** データをクラウドに送信
4. **Part 4:** データを分析し、意味を読み取る

これらを統合して、完全な IoT システムを構築しました！

🎯 学生に伝えるべき重要なメッセージ：

- 「皆さんは今、**IoT システムを作れる技術者**です」
- 「センサー、Wi-Fi、クラウド、分析。これらを**つなげる力**を身につけました」
- 「このシステムは、スマート農業、スマートホーム、環境モニタリングなど、**実社会で使われている技術**そのものです」
- 「ここで学んだことを、**次の学び、次のプロジェクト**に活かしてください」

🔧 事前準備チェックリスト

- 24 時間分のサンプルグラフを準備（説明用）
- グループワーク用のワークシート準備（任意）
- 発表時間を確保（学生の発見を共有）
- 実習 2 全体の振り返りスライド準備

§ 指導のコツ 💡

1. 学生の発見を称賛する

グラフから何か面白いことを見つけた学生がいたら、大いに褒めましょう。「良い観察ですね！」の一言が、分析意欲を高めます。

2. 「答え」を教えすぎない

データ分析には「正解」がない場合も多いです。学生の解釈を尊重し、考えるプロセスを大切にしましょう。

3. 実社会とのつながりを示す

学生が作ったシステムが、実際の IoT サービスと同じ仕組みであることを強調しましょう。

4. 達成感を最大化する

Part 4 の最後は、実習 2 全体の達成を祝う時間です。学生の努力を認め、達成感を共有しましょう。

§ Part 4 および実習 2 全体の成功基準 🎯

- ✅ 学生がグラフから意味を読み取れている
- ✅ データ分析の 4 ステップを理解している

- ✅ システム改善のアイデアを考えることができる
- ✅ 実習 2 全体の流れを説明できる
- ✅ **IoT システムを作り上げた達成感を感じている**（最重要）

IoT 技術応用
実習 3 システム統合
導入編 教員用指導ガイド

§ 🎯 導入セッションの目的

- 実習 3 の全体像を明確に伝え、学習意欲を高める
- 新しいツール（Python、Node-RED）への不安を解消
- 事前準備の状況を確認し、スムーズなスタートを切る
- 実習 1・2 との連続性を意識させ、学習の積み上げを実感させる

§ 📋 事前準備チェックリスト

教員側の準備

✅ **ソフトウェア環境**

- Anaconda（Python 3.x）のインストール確認
- Node.js、Node-RED のインストール確認
- 教員用 PC で実習 3 の全 Part を事前に実行し、動作確認
- node-red-dashboard の動作確認

✅ **デモ用資料**

- 完成した IoT システムのデモ動画（推奨）
- システム構成図（ESP32→MQTT→Node-RED→Python）
- 実習 1・2・3 の比較表スライド
- Node-RED ダッシュボードのスクリーンショット

✅ **ネットワーク環境**

- Wi-Fi 接続の安定性確認
- MQTT ブローカーへのアクセス確認（test.mosquitto.org）
- 学内 MQTT ブローカーがあれば、そちらを推奨

学生への事前指示（1 週間前）

必須インストール

- Anaconda（約 600MB、インストールに 30 分程度）
  → https://www.anaconda.com/download
- Node.js（約 50MB）
  → https://nodejs.org/
- Node-RED（Node.js インストール後、コマンド: npm install -g node-red）

動作確認

- Jupyter Notebook が起動するか（Anaconda Navigator 経由）
- Node-RED が起動するか（コマンド: node-red）
- ESP32 とセンサーが正常動作するか（実習 2 の継続）

§ ポイント

内容	指導のポイント
オープニング	実習 3 が「総仕上げ」であることを強調
実習 1・2 の振り返り	学習の積み上げを視覚的に示す（比較表）
新ツール紹介（Python、Node-RED）	実際の画面を見せて興味を引く
システム全体像の説明	データの流れを図で明示
事前準備確認	挙手で確認、未準備者をリストアップ

§ ❗ よくあるトラブルと対処法

1. Anaconda のインストールが終わらない

原因： サイズが大きく、回線が遅い

対処：

- 事前インストールを強く推奨（授業中は間に合わない）
- 未インストールの学生には、Part 1 の間にバックグラウンドでインストールさせる
- それでも間に合わない場合、Google Colab の利用を提案

2. Node-RED が起動しない

原因： Node.js が正しくインストールされていない、またはポート競合

対処：

- Node.js のバージョン確認：node -v
- ポート 1880 が使用中の場合：他のアプリを終了、または別ポートで起動
- node-red -p 1881 で別ポートを指定

3. ESP32 が動作しない

原因： 実習 2 から時間が経過し、配線を忘れた、またはセンサー故障

対処：

- 実習 2 の配線図を再確認
- シリアルモニタでエラーメッセージを確認
- 予備の DHT22 センサーを準備しておく

4. 実習 2 のデータが蓄積されていない

原因： 実習 2 で Ambient/ThingSpeak への送信ができていなかった

対処：

- サンプル CSV ファイルを配布（教員が事前に用意）
- または、教員の Ambient データをダウンロードさせる

§ 💡 指導上の工夫

1. **デモの効果的な活用**
   - 完成したシステムのデモを**最初に見せる**ことで、ゴールイメージを明確に
   - Jupyter Notebook と Node-RED の画面を実際に開いて見せる
   - 「コードを書かなくても、ブロックをつなぐだけで動く」ことを Node-RED で実演
2. **不安の解消**
   - Python が初めての学生が多いため、「難しくない」ことを強調
   - 「C++より書きやすい」「Jupyter Notebook が親切」とポジティブに伝える
   - 「わからなかったら、いつでも質問してください」と安心感を与える
3. **モチベーションの向上**
   - 実習 1 からの成長を振り返らせる（「LED から始めて、今はシステム構築まで！」）
   - 実社会での応用例を具体的に紹介（農業、工場、医療など）
   - 「これができれば、就職でアピールできる」と実利的な面も伝える

§ 📝 次の準備

導入が終わったら、**Part 1** にスムーズに移行するため：

- 全員が Jupyter Notebook を起動できているか確認
- CSV ファイルのダウンロード方法を確認（Ambient/ThingSpeak）
- Part 1 のワークブックを配布（印刷 or PDF）

IoT 技術応用
実習 3 システム統合
Part 1 ：Python とデータ分析の基礎 教員用指導ガイド

§ Part 1 の指導目標🎯

- 全員が Jupyter Notebook を起動・操作できるようになる
- Pandas でデータを読み込み、表示できる
- 基本統計量の意味を理解する
- Python への苦手意識を払拭する

§ 内容項目

内容	重要度
Part 1 の概要説明	★★★
Jupyter Notebook 起動と基本操作	★★★
Pandas でデータ読み込み	★★★
基本統計量の確認	★★☆

§ よくあるトラブルと対処法❗

1. Jupyter Notebook が起動しない

症状： Anaconda Navigator で「Launch」を押しても何も起こらない
原因：

- Anaconda が正しくインストールされていない
- ブラウザが自動起動しない設定になっている
- 既に起動している（ブラウザのタブを確認）

対処法：

1. コマンドライン起動を試す: jupyter notebook
2. 表示された URL を手動でブラウザに入力
3. それでもダメなら、Google Colab を使用（colab.research.google.com）

2. CSV ファイルが読み込めない（FileNotFoundError）

症状： pd.read_csv()実行時にエラー
原因： ファイルパスが間違っている
対処法：

1. CSV ファイルの場所を確認（デスクトップ、ダウンロードフォルダなど）
2. フルパスで指定: r'C:¥Users¥username¥Desktop¥data.csv'
3. Jupyter Notebook と同じフォルダに移動させる
4. または、サンプル CSV を配布

3. 日本語が文字化けする

症状： CSV の日本語列名が「?」や「■」で表示される
原因： 文字コードの問題
対処法：

```
df = pd.read_csv('data.csv', encoding='utf-8')
# それでもダメなら
df = pd.read_csv('data.csv', encoding='shift-jis')
```

4. セルが実行されない

症状： Shift+Enter を押しても何も起こらない
原因： カーネルがビジー状態、またはクラッシュ
対処法：

- カーネルを再起動: 「Kernel」→「Restart」
- セルの左側が「*」になっている場合は、実行中なので待つ

5. Pandas がインポートできない（ModuleNotFoundError）

症状： import pandas as pd でエラー
原因： Anaconda を使っていない、または環境が壊れている
対処法：

1. Anaconda Navigator から起動しているか確認
2. ターミナルで: conda install pandas
3. それでもダメなら、Anaconda の再インストール

§ 指導上の工夫とポイント 💡

1. Jupyter Notebook の第一印象が重要

- 「Hello, Python!」の実行で成功体験を与える
- 「C++より簡単」「すぐに結果が見える」とポジティブに強調
- セルの追加・削除・実行を丁寧に実演
- Shift+Enter のショートカットを繰り返し伝える

2. データ読み込みの成功率を上げる

- 事前にサンプル CSV を配布しておく（確実に読み込める状態）
- ファイル名を統一（例: ambient_data.csv）
- パスの書き方を複数パターン紹介:
    - 相対パス: 'data.csv'
    - 絶対パス（Windows）: r'C:¥Users¥...'
    - 絶対パス（Mac）: '/Users/...'

3. Pandas の便利さを実感させる

- df.head()でデータが表形式で綺麗に表示されることを強調
- df.describe()の出力を見せて、「たった 1 行でこれだけの統計が！」と驚かせる
- Excel との比較: 「Excel でやったら手作業で大変だけど、Pandas なら一瞬」

4. 統計用語の説明（簡潔に）

- **平均（mean）**：データの合計÷個数
- **中央値（median）**：データを並べた時の真ん中の値
- **標準偏差（std）**：データのバラつき具合（大きい＝バラバラ、小さい＝まとまっている）
- **最大・最小（max/min）**：そのまま

※深く理解させるより、「こういう指標がある」程度で OK

§ 進捗確認のチェックポイント✅

Part 1 終了前に、以下を全員が達成しているか確認：

項目	確認方法
Jupyter Notebook 起動	画面共有で確認、または巡回
Pandas インポート	「実行できた人？」と挙手確認
CSV 読み込み成功	df.head()の出力が見えるか確認
統計量表示	df.describe()が実行できたか

📝 次回（Part 2）への準備

- Part 1 で読み込んだデータをそのまま使うため、ノートブックを保存させる
- 「Ctrl+S」または「File」→「Save」で保存
- 次回は Matplotlib を使うため、必要なら事前インストール（通常は Anaconda に含まれる）

§ 学生へのフィードバック例🎓

Part 1 終了時に伝えると良いこと：

- 「お疲れさまでした。Python の第一歩が踏み出せました。」
- 「データが読み込めた瞬間、『おお！』という声が聞こえました。それが大事です。」
- 「次の Part 2 では、このデータをグラフにします。もっと面白くなりますよ。」
- 「もしつまずいても、焦らないでください。Python は慣れです。」

IoT 技術応用
実習 3 システム統合
Part 2 ～ 4　教員用指導ガイド

📚 本マニュアルの構成

Part 2-4 の指導ポイントを効率的にまとめています。各 Part の詳細は個別ワークブックを参照してください。

§ Part 2 ：IoT データの高度な分析📊

Matplotlib、統計分析、異常値検出、予測モデル

§ Part 2 の指導目標🎯

- Matplotlib でグラフを作成できる
- 統計分析（相関、標準偏差）の意味を理解する
- 異常値検出の概念を学ぶ
- 予測モデルの基礎に触れる

§ よくあるトラブル❗

1. グラフが表示されない

原因: %matplotlib inline が実行されていない
対処: ノートブックの最初で実行させる

2. 日本語が文字化け（□□□で表示）

原因: フォント設定の問題
対処:

```
plt.rcParams['font.sans-serif'] = ['MS Gothic', 'Yu Gothic', 'Hiragino Sans']
plt.rcParams['axes.unicode_minus'] = False
```

3. 相関係数の解釈がわからない

対処: 散布図を見せながら視覚的に説明

- 1 に近い → 右上がりの点の並び
- -1 に近い → 右下がりの点の並び
- 0 に近い → 点がバラバラ

§ 指導のポイント💡

- **グラフの威力:** 数字の羅列よりも一目でパターンがわかることを強調
- **統計用語:** 深く理解させるより、「こういう指標がある」程度で OK
- **異常値検出:** 3σ 法の数式より、「平均から大きく外れた値」という直感的理解を優先
- **予測モデル:** 「機械学習の入口」であることを伝え、興味を引く

§ 進捗確認✅

- ☐ グラフが正常に表示される
- ☐ 相関係数を計算できた
- ☐ 異常値検出を実行できた

§ Part３：Node-RED による IoT フロー構築🔁

ビジュアルプログラミング、MQTT、ダッシュボード

§ Part３の指導目標🎯

- Node-RED を起動・操作できる
- MQTT でデータ送受信できる
- ダッシュボードを作成できる
- データ処理フローを構築できる

§ よくあるトラブル❗

1. Node-RED が起動しない

原因: Node.js が未インストール、またはポート競合
対処:

- Node.js バージョン確認: node -v
- 別ポートで起動: node-red -p 1881

2. MQTT ブローカーに接続できない

原因: ブローカーアドレスやポートの設定ミス
対処:

- test.mosquitto.org:1883 を確認
- 学内に MQTT ブローカーがあればそちらを推奨
- mqtt in ノードの状態を確認（「接続済み」と表示されるべき）

3. ダッシュボードが表示されない

原因: node-red-dashboard が未インストール
対処:

- 「パレットの管理」→「ノードを追加」

- 「node-red-dashboard」を検索してインストール
- インストール後、Node-RED の再起動が必要な場合あり

4. データが受信できない

原因: トピック名の不一致
対処:

- 送信側と受信側のトピック名を厳密に確認（大文字小文字も区別）
- debug ノードでデータの流れを確認

§ 指導のポイント💡

- **視覚的な理解:** ノードをつなぐだけでシステムができる、という直感性を強調
- **デプロイの重要性:** 変更したら必ずデプロイすることを徹底
- **デバッグの習慣:** debug ノードを活用し、データの流れを可視化する習慣をつけさせる
- **ダッシュボードの感動:** 初めて表示された瞬間を盛り上げる

§ 進捗確認✅

- ☐ Node-RED が起動している
- ☐ MQTT で送受信できた
- ☐ ダッシュボードが表示された
- ☐ gauge、chart、text が動作している

§ Part４：統合 IoT システムの開発🌐

ESP32 + Python + Node-RED の完全統合

§ Part４の指導目標🎯

- ESP32 から MQTT でデータ送信できる
- Node-RED でリアルタイム表示できる
- Python から分析結果を送信できる
- アラート機能を実装できる
- **システム全体が連携して動作する**

§ よくあるトラブル❗

1. ESP32 が MQTT に接続できない

原因: Wi-Fi 接続失敗、またはブローカー設定ミス
対処:

- シリアルモニタでエラーメッセージ確認
- Wi-Fi SSID・パスワードの再確認

- ブローカーアドレスの確認（test.mosquitto.org）

2. PubSubClient ライブラリがない

対処:

- Arduino IDE →「スケッチ」→「ライブラリを管理」
- 「PubSubClient」を検索してインストール

3. Node-RED で ESP32 のデータが受信できない

原因: トピック名の不一致、または ESP32 が送信していない
対処:

- ESP32 のシリアルモニタで「MQTT 送信完了」を確認
- Node-RED の mqtt in ノードで正しいトピックを購読しているか確認
- debug ノードで受信状況を確認

4. Python から paho-mqtt が import できない

対処:

- Jupyter Notebook で: !pip install paho-mqtt
- 実行後、カーネルを再起動

5. システム全体が連携しない

原因: 複数箇所でエラーが発生している
対処（デバッグの順序）:

1. ESP32 → シリアルモニタで送信を確認
2. Node-RED → debug ノードで受信を確認
3. ダッシュボード → ブラウザで表示を確認
4. Python → 送信メッセージを確認
5. Node-RED → 分析結果の受信を確認

1 つずつ確認し、どこで止まっているか特定する

§　指導のポイント 💡

- **統合の難しさ:** 複数技術を組み合わせるため、エラーが出やすいことを事前に伝える
- **段階的確認:** 1 つずつ動作確認しながら進める（焦らせない）
- **達成感の演出:** システムが動いた瞬間を最大限盛り上げる（拍手など）
- **個別差への対応:** 早く終わった学生には発展課題、遅れている学生には個別サポート
- **記録の推奨:** 完成したダッシュボードのスクリーンショットを撮らせる

§　最終確認（Part４完了時）✅

全員が以下を達成しているか確認：

項目	確認方法
ESP32→Node-RED	ダッシュボードがリアルタイム更新される
ESP32→Ambient	Ambient グラフが更新される
Python→Node-RED	分析結果がダッシュボードに表示される
アラート機能	閾値を超えると通知が出る

§ 実習 3 全体の振り返り🎓

Part 4 完了後、以下を学生と共有すると効果的：

- 「実習 1 から実習 3 まで、どれだけ成長したか」を振り返る
- 完成したシステムの**実社会での応用例**を紹介（農業、工場、医療など）
- 今後の学習の方向性（機械学習、クラウド、セキュリティなど）を示す
- 学生の努力を称賛し、自信を持たせる

§ 確認項目📝

項目	確認方法
Python でデータ読み込み	ワークブック記入、またはノートブック提出
グラフ作成と分析	作成したグラフの提出
Node-RED フロー構築	フローのスクリーンショット
統合システム完成	ダッシュボードの動画または画像
考察・レポート	ワークブックの記述内容

§ 提出物の管理💾

学生に提出させるもの

- 完成した Jupyter Notebook（.ipynb ファイル）
- Node-RED のフロー（JSON エクスポート、またはスクリーンショット）
- ダッシュボードのスクリーンショットまたは動画
- ESP32 の Arduino コード（.ino ファイル）
- ワークブック（記入済み PDF または印刷物）

提出方法の例

- LMS（学習管理システム）へのアップロード
- Google Drive などでの共有
- GitHub リポジトリ（発展的な学生向け）

令和7年度「地方やデジタル分野における専修学校理系転換等推進事業」

情報成長分野の教育プログラム整備と教員育成による学科転換・新設推進事業

教員研修資料

令和８年２月

一般社団法人全国専門学校情報教育協会

〒164-0003 東京都中野区東中野 1-57-8 辻沢ビル３F

電話：03-5332-5081 FAX.03-5332-5083

●本書の内容を無断で転記、掲載することは禁じます。