学科	ITエンジニア科3年制	シラバス
コード		科目趣旨・目的（背景となる社会情勢・業界動向・資格試験との関係、および、カリキュラム全体における位置づけ）
年度	2022	近年、AIが脚光を浴びており、AIを支える機械学習技術およびデータ分析技術が急速に発展・普及している。機械学習やデータ分析を活用する多くのプログラムはPython言語で記述されているため、産業界においてPython言語のスキルを有した技術者の確保が課題となっている。また、Pythonは比較的なシンプルな言語仕様から構成されており、元々、海外では初学者向けのプログラミング言語として人気があった。以上の背景を踏まえて、本科目ではPython言語を通してプログラミングの基礎を習得して、さらに、Python言語の言語仕様および代表的な機能および標準ライブラリを理解することを目標とする。なお、Python言語は動的型付け言語であり、かつ、オブジェクト指向プログラミングに対応した言語である。したがって、本科目では静的型付け言語と比較すると、型に関する学習時間は短くなっており、その分、Python特有の言語機能やオブジェクト指向プログラミング、アルゴリズム、ソフトウェア品質など、Pythonエンジニアに求められる幅広い知識を扱う。型に関する知識を補うためには、別の科目において、Javaなどの静的型付け言語にも触れることが望ましい。
学年	2	
期	後期	
分野名	基礎	
科目名	**Python基礎**	
単位	2	
授業形態		科目概要
実務連携型授業	○	本科目の前半では、コンピュータに関する基礎知識から始まり、変数・代入・各種演算子・制御構文（条件分岐および繰り返し）など、プログラミングの基礎的な知識を学ぶ。その後、Pythonがサポートする様々なコレクションを通して、関数・クラス・オブジェクトなど、プログラムのモジュール化に関する基礎的な知識を学ぶ。後半では、Pythonの標準ライブラリの紹介や、ソフトウェア品質、Pythonにおけるコーディング規約など、実務において必要とされる応用的な知識を学ぶ。その後、計算量の概念を学び、代表的なアルゴリズムの学習やゲーム開発を通して、複雑な処理を有するプログラムを記述するスキルを鍛錬する。
必修・選択	必修	
前提とする科目		
展開科目		キーワード
関連資格		制御フロー　関数　クラス　コレクション　モジュール　入出力　例外処理　コーディング規約　ソートアルゴリズム　探索アルゴリズム
教員	常勤	到達目標
		コンピュータやプログラミングの基礎的な知識を理解して、また、Python特有の機能（コレクション・モジュール・標準ライブラリ）を理解することで、Pythonの公式ドキュメントを独力で読んで内容を理解できるようにすることを目標とする。

コマシラバス（90分授業コマ単位のシラバス）

90分/コマ	コマ主題	コマシラバス項目		内容
1	プログラミング・Pythonの基本知識	1_1	コマ主題細目	①コンピュータとソフトウェア　②プログラム　③プログラミング言語の種類　④Python言語
		1_2	細目レベル	①コンピュータについて、主要装置（CPU、メモリ、ハードディスク、キーボード、モニター、プリンター等）を図示して、各装置の機能および役割を解説する。ソフトウェアについて、オペレーティングシステム（OS）・インタプリタ・アプリケーション・ライブラリなど、ソフトウェアを実行する際の階層構造を図示して、各階層の詳細を説明する。さらに、OSやアプリケーション・Webアプリケーション（Webアプリ）・ミドルウェアなど、いくつかのソフトウェアの種類ついて図示しながら解説する。 ②ソースコード・機械言語・アセンブリ言語などプログラムの設計情報とコンパイラやインタプリタなどの処理系について解説した上で、コンピュータがプログラムを読み取って実行するまでの流れを図示しながら解説する。 ③構成部品、および、各構成部品の関係性を示した上で、各部品の開発に利用可能なプログラミング言語の特徴を解説する。例えば、Webアプリの場合は、クライアントプログラムを記述する言語とて、HTML・CSS・JavaScriptについて解説した上で、JavaScriptにトランスパイル可能な言語およびJavaScriptフレームワーク（React・Vue.js・AngularJS）について解説する。また、サーバプログラムを記述する言語として、Java・PHP・Python・JavaScriptについて解説する。同様にパソコンやスマートフォン向けのアプリケーションやスクリプトを記述するための言語についても合わせて解説する。 ④一般的なアプリケーションの開発においてPythonを使用する際と、機械学習・データ分析においてPythonを使用する際の相違点について解説した上で、各場面で利用するライブラリおよびフレームワークについて解説する。また、Jupyter NotebookおよびGoogle Colabなどの実行環境について解説する。
		1_3	5キーワード	①プログラム　②プログラミング言語　③Python言語　④データ分析　⑤実行環境
		1_4	コマ要素	□実務連携　□理解度確認テスト　■オリジナル教材　■ICT活用　■実習・実技・実験・演習　□該当なし

2	変数・データ型・代入・数値計算	2_1	コマ主題細目	①変数　②代入　③データ型　④演算子と数値計算　⑤プログラム作成課題
		2_2	細目レベル	①変数とは、プログラムで使用するデータに名前をつけて保持しておくための概念であることを解説する。また、変数の使い方について、変数を定義して値を読み書きするサンプルプログラムを提示しながら解説する。 ②代入とは、変数に新たなデータを格納する処理であることについて、サンプルプログラムを提示しながら解説する。また、代入文は、「=」の記号を用いて「(変数) = (値)」と記述することを解説する。 ③基本的なデータ型であるint型、float型、str型、bool型について、サンプルプログラムを提示しながら解説した上で、コレクションのデータ型(list型、tuple型、set型、dict型)について簡単に解説する。 ④演算子の一覧、計算時の優先順位、int型とfloat型の数値計算について、サンプルプログラムを提示しながら解説する。
		2_3	5キーワード	①変数　②演算子　③代入　④データ型　⑤数値計算
		2_4	コマ要素	□実務連携　□理解度確認テスト　■オリジナル教材　■ICT活用　■実習・実技・実験・演習　□該当なし
3	制御フロー	3_1	コマ主題細目	①リストの基礎　②繰り返し (1)　③条件分岐　④繰り返し (2)　⑤繰り返しの制御　⑥パターンマッチ
		3_2	細目レベル	①リストの基礎について解説する。リストリテラルの表記方法、リスト内の各要素へのアクセス方法について、サンプルプログラムを提示しながら解説する。 ②for文の構文と意味を説明した上で、for文とrangeクラスを組み合わせた繰り返し方法について解説する。また、for文はリストなどの指定した要素集合の各要素を列挙しながら、処理を繰り返すための構文であることについて、サンプルプログラムを提示しながら解説する。 ③if文およびif式の構文および意味を解説する。また、if文およびif式を用いることで、指定した条件を満たしたときにだけ、特定の処理を実行できることについて、サンプルプログラムを提示しながら解説する。 ④while文を用いた繰り返しについて解説する。while文は指定した条件を満たす間、処理を繰り返すための構文であることについてサンプルプログラムを提示しながら解説する。 ⑤無限ループについて解説した後に、break文およびcontinue文を扱った繰り返し処理の制御方法について、サンプルプログラムを提示しながら解説する。 ⑥match 文について解説する。match 文はPython3.10 でサポートされた新機能であり、パターンごとに分岐を記述でき、マッチしたパターンの処理だけを実行できることについて、サンプルプログラムを提示しながら解説する。また、パターンマッチ特有の機能であるワイルドカードについても、サンプルプログラムを提示しながら解説する。
		3_3	5キーワード	①リスト　②for文　③if文　④while文　⑤match 文
		3_4	コマ要素	□実務連携　□理解度確認テスト　■オリジナル教材　■ICT活用　■実習・実技・実験・演習　□該当なし
4	関数とメソッド	4_1	コマ主題細目	①関数　②オブジェクトとメソッド
		4_2	細目レベル	①関数とその使用法を解説する。まず、電車を表示するサンプルプログラムを用いながら、関数の定義・機能と関数を用いるための要件である「定義」および「呼び出し」を説明する。次に、仮引数及び実引数の定義を紹介する。電車を表示するサンプルプログラムを参照しつつ、仮引数を関数定義に使用することによってより複雑なプログラムを作ることが可能となることを説明する。関数定義において仮引数を使用する場合の注意点についても解説する。 ②文字列・リスト・関数といったオブジェクトは、メソッドを有する。本講義では、まず、オブジェクトのID(本講義では、オブジェクトの同一性をこのように呼称する)をis演算子または「= =」を用いて比較することができることについて、文字列を例にとって確認し、upper()、lower()といった文字列のメソッドをいくつか紹介していく。これまでの講義と同様、それらの説明はサンプルプログラムを用いて行う。次に、リストのメソッドについて解説する。具体的には、append()、extend()、reverse()について、サンプルプログラムを使用して説明をする。リストのメソッドと文字列のメソッドとの間には、いくつかの相違が見られるため、学習者はその点についても把握しなければならない。最後に、関数をオブジェクトとして使用した場合の挙動について、サンプルプログラムを用いて解説する。
		4_3	5キーワード	①関数　②引数　③戻り値　④オブジェクト　⑤メソッド
		4_4	コマ要素	□実務連携　□理解度確認テスト　■オリジナル教材　■ICT活用　■実習・実技・実験・演習　□該当なし

5	コレクション	5_1	コマ主題細目	①コレクションの種類　②リスト　③タプル　④集合　⑤辞書
		5_2	細目レベル	①リスト、辞書、タプルの概要について解説する。いずれも複数のデータを保持するためのデータ構造であることは共通しているが、データの保持の仕方に違いがあり、置換、削除、挿入、検索、走査などの処理の性能に違いがあることについて、サンプルプログラムを提示しながら解説する。 ②リストとは、複数のデータに番号を付与して特定の順番で並べたデータ構造であることについて、サンプルプログラムを提示しながら解説する。 ③タプルとは、ひとかたまりのデータを組みとして保持するデータ構造であることを解説する。 タプルはリストとよく似ているが、リストは複数のデータの繰り返しに用いることが通常であり、タプルは全体が一つのデータであって、各要素をデータの属性を表すために用いることについて、サンプルプログラムを提示しながら解説する。 ④集合とは、重複を許さず、また、順序を保持せずに複数の値の集合を記録するデータ構造であることについて、サンプルプログラムを提示しながら解説する。 ⑤辞書とは、複数のデータをキー（データ検索のID）とバリュー（実際のデータ）の対応関係で保持するデータ構造であることについて、サンプルプログラムを提示しながら解説する。
		5_3	5キーワード	①リスト　②タプル　③集合　④辞書　⑤キー/バリュー
		5_4	コマ要素	□実務連携　□理解度確認テスト　■オリジナル教材　■ICT活用　■実習・実技・実験・演習　□該当なし
6	クラス	6_1	コマ主題細目	①名前空間とスコープ　②クラス　③オブジェクト指向プログラミング　④継承
		6_2	細目レベル	①Pythonにおいて、クラスは名前空間の仕組みを活用して実現される。そこで、まず、名前空間およびそれと関連するスコープについて解説する。 ②クラスの定義（コンストラクタ・インスタンス変数・インスタンスメソッド）と定義したクラスのインスタンスの生成方法を解説する。また、Pythonの組み込み型であるリスト・タプル・集合・辞書にもクラスがあり、各組み込み型のリテラルはインスタンス生成の簡略表記に相当することを解説する。 ③オブジェクト指向プログラミングとは、プログラムをオブジェクト（データと処理の組み合わせ）という概念に基づいて整理して部品化していく考え方をいう。本講義では、人間のBMI値を計算する場面を想定し、オブジェクト指向プログラミングの考え方に基づく場合と従来の手続型プログラミングの考え方に基づく場合とを比較しつつ、オブジェクト指向プログラミングの有用性を理解していく。 ④クラスの継承の方法および意味論とともに、継承によるモジュール化のメリットを解説する。さらに、オブジェクト間の継承関係を確認する方法について、isinstance関数とissubclass関数を扱ったサンプルコードを提示しながら解説する。また、多重継承について、基本的な概念のみを解説する。
		6_3	5キーワード	①オブジェクト指向プログラミング　②クラス　③インスタンス　④継承によるモジュール化　⑤多重承継
		6_4	コマ要素	□実務連携　□理解度確認テスト　■オリジナル教材　■ICT活用　■実習・実技・実験・演習　□該当なし
7	モジュール	7_1	コマ主題細目	①モジュール　②パッケージ　③標準ライブラリと外部ライブラリ
		7_2	細目レベル	①モジュール・パッケージ・ライブラリの特徴および相違点について解説する。 モジュールとは、1つのPythonファイルのことであることを解説する。パッケージとはPythonファイルを集めてディレクトリとし、外部に公開したものであることを解説する。ライブラリはパッケージをWeb上に公開し、誰でも利用可能なものとして保守・運用されているものであることを解説する。 モジュールの作り方、呼び出し方について、サンプルプログラムを提示しながら解説する ②パッケージの作り方、呼び出し方について解説する。 通常のディレクトリをPythonパッケージとして認識させるための _init_.py について、サンプルプログラムを提示しながら解説する。さらに、import 文の構文と意味を解説してから、絶対インポートと相対インポートを解説する。*を用いた全インポートについて、サンプルプログラムを提示しながら解説する。 ③標準ライブラリと外部ライブラリの違いと、具体例について解説する。 標準ライブラリとは、Python が公式に定めた、処理系が必ずサポートしなければならないライブラリのことであると解説する。一方、外部ライブラリとは、Pythonコミュニティのプログラマが独自に作成しているライブラリのことであると解説する。 Pythonは外部ライブラリが非常に充実しており、複雑な処理を初心者でも簡単に試用できることが大きな特徴であると述べる。
		7_3	5キーワード	①モジュール　②パッケージ　③標準ライブラリ　④外部ライブラリ　⑤import 文
		7_4	コマ要素	

8	入出力	8_1	コマ主題細目	①標準入出力　②ファイル入出力　③クリーンアップ処理
		8_2	細目レベル	①標準入出力および標準エラー出力の概念について解説した上で、Pythonにおける標準入出力および標準エラー出力の方法について、そこで使用される関数(input()関数、print()関数、strip()関数等)に関する一般的な説明とともに、サンプルプログラムを提示しながら解説する。 ②ファイルの入出力について、ファイルの概念に関する説明及びその具体例を交えながら解説する。その上で、Pythonにおいてファイルを開き、文字列データおよびバイナリデータを読み書きする方法について、サンプルプログラムを提示しながら解説する。 ③ファイルなどのリソースは、プログラムが本来期待される動作と異なる挙動をすることを防止するため、その使用後に解放する必要があることを解説する。その上で、開いたファイルを閉じてリソースを解放しプログラムが意図した動作をするようにする方法として、close()関数を使う方法とwith 文を使う方法を紹介する。その後、それらの方法について、サンプルプログラムを提示しながら、close()関数及びwith文の使用に際しての注意点も指摘しながら解説する。
		8_3	5キーワード	①標準入出力　②標準エラー出力　③ファイル入出力　④with文　⑤closeメソッド
		8_4	コマ要素	□実務連携　□理解度確認テスト　■オリジナル教材　■ICT活用　■実習・実技・実験・演習　□該当なし
9	エラーと例外	9_1	コマ主題細目	①エラーの種類　②例外の処理　③例外の送出　④独自例外の定義
		9_2	細目レベル	①Pythonには構文エラーと例外の2種類のエラーがあることを解説して、各種類のエラーの具体例やエラーから問題を突き止める方法について、サンプルプログラムとエラーの例を提示しながら解説する。 ②try文、try節、except節、catch節、finally節の構文および意味を解説して、例外が送出されたときの対応方法を解説する。また、いくつかの具体的な例外を取り上げ、各例外の意味について、例外を処理するサンプルプログラムを提示しながら解説する。 ③raise文の構文および意味を解説して、例外の送出方法を解説する。また、fromを用いて例外を連鎖させる方法を解説する。さらに、いくつかの具体的な例外を取り上げ、各例外の使い方について、例外を創出するサンプルプログラムを提示しながら解説する。 ④既存の例外クラスを継承したクラスを定義することで、独自の例外を定義する方法を解説する。親クラスとなるいくつかの具体的な例外を取り上げ、各例外を継承して独自の例外を定義する方法について、サンプルプログラムを提示しながら解説する。
		9_3	5キーワード	①構文エラー　②例外　③例外処理　④raise文　⑤独自の例外
		9_4	コマ要素	□実務連携　□理解度確認テスト　■オリジナル教材　■ICT活用　■実習・実技・実験・演習　□該当なし
10	ソフトウェア品質とコーディング規約	10_1	コマ主題細目	①ソフトウェア品質　②コーディング規約
		10_2	細目レベル	①ソフトウェア品質は、利用者がソフトウェアを使用する際に操作するユーザインタフェースに関する品質から、開発者がソフトウェアを保守する品質まで、幅広い内容を包含する概念である。本講義では、ソフトウェア品質を定量的に評価するための枠組みとなるソフトウェア品質モデルおよびそこで示される各品質特性について概観する。JIS X 25010:2013およびISO/IEC 25010:2011（SQuaREシリーズ）において、ソフトウェア品質モデルは製品品質モデル、利用時の品質モデル、データ品質モデルの3種類に分類されており、本講義ではその中でもプログラムに関連する製品品質モデル及び利用時の品質モデルについて解説する。学習者には、ソフトウェア品質の概要を把握するとともに、プログラムの作成に密接に関わる品質特性である機能適合性、性能効率性、保守性に関し深く理解することが求められる。 ②コーディング規約とは、プログラムを記述する際に守るべきルールである。授業内では、コーディング規約の一例としてPythonにおいて最も著名なコーディング規約であるPEP 8の中からインデント、二項演算子の改行位置、式や文中の空白文字、コメント、命名規約を抜粋して解説する。また、コーディング規約に違反した場合の具体例を示し、その違反の影響を考えてもらい、コーディング規約に反した場合にプログラムの内容を理解することが難しくなることを体感する。コーディング規約の存在意義、遵守の必要性、コーディング規約とソフトウェア品質の関係性等を理解していく。
		10_3	5キーワード	①ソフトウェア品質　②品質特性　③品質副特性　④PEP　⑤コーディング規約
		10_4	コマ要素	□実務連携　□理解度確認テスト　■オリジナル教材　■ICT活用　■実習・実技・実験・演習　□該当なし

11	探索アルゴリズム	11_1	コマ主題細目	①時間計算量・空間計算量　②線形探索　③二分探索
		11_2	細目レベル	①時間計算量と空間計算量は、アルゴリズムの性能を評価する上で重要な概念である。まずアルゴリズムの定義を説明したうえで、アルゴリズムの性能を評価する方法として時間計算量と空間計算量の定義を説明する。さらに、計算量は関数の主要項によって支配的に決定されることを説明した後に、計算量を近似する方法としてO記法を解説する。加えて、計算量をO記法で表したときの式の形によって、データサイズによる計算量増加の特性が変わることを説明する。 ②探索アルゴリズムの概要について確認したうえで、探索アルゴリズムの一つとして、リストや配列などに入った要素を先頭から1つずつ確認し探索を行うアルゴリズムである線形探索について解説する。同アルゴリズムの特徴と抽象化された手順を説明した後に、Pythonによる実装とその解説によりアルゴリズムの動作を理解させる。演習として、同アルゴリズムの動作過程を出力するPythonプログラムを実装させる。 ③探索アルゴリズムの一つとして、整列済みの配列などに対して探したい要素との大小関係をもとに探索を行うアルゴリズムである二分探索について解説する。同アルゴリズムの特徴と抽象化された手順を説明した後に、Pythonによる実装とその解説によりアルゴリズムの動作を理解させる。演習として、同アルゴリズムの動作過程を出力するPythonプログラムを実装させる。
		11_3	5キーワード	①時間計算量　②空間計算量　③探索アルゴリズム　④線形探索　⑤二分探索
		11_4	コマ要素	□実務連携　□理解度確認テスト　■オリジナル教材　■ICT活用　■実習・実技・実験・演習　□該当なし
12	ソートアルゴリズム	12_1	コマ主題細目	①バブルソート　②選択ソート　③挿入ソート　④マージソート
		12_2	細目レベル	①初めに、ソート（並べ替え）アルゴリズムの概要に触れ、ソートの順序を決定する方法である昇順・降順の意味について確認する。簡単なソートアルゴリズムの一つとして、隣り合う要素の大小を比較しながらソートを行うアルゴリズムであるバブルソートについて解説する。同アルゴリズムの特徴と抽象化された手順を説明した後に、Pythonによる実装とその解説によりアルゴリズムの動作を理解させる。演習として、同アルゴリズムの動作過程を出力するPythonプログラムを実装させる。 ②簡単なソートアルゴリズムの一つとして、配列内の最大値もしくは最小値を見つけてソートを行うアルゴリズムである選択ソートについて解説する。同アルゴリズムの特徴と抽象化された手順を説明した後に、Pythonによる実装とその解説によりアルゴリズムの動作を理解させる。演習として、同アルゴリズムの動作過程を出力するPythonプログラムを実装させる。 ③簡単なソートアルゴリズムの一つとして、追加要素を整列済みの配列のどこに入るか調べて挿入を行うことでソートを行うアルゴリズムである挿入ソートについて解説する。同アルゴリズムの特徴と抽象化された手順を説明した後に、Pythonによる実装とその解説によりアルゴリズムの動作を理解させる。演習として、同アルゴリズムの動作過程を出力するPythonプログラムを実装させる。 ④高速なソートアルゴリズムの一つとして、ソート対象のデータを分割していき、細分化を終えたら整列しながら要素を併合していくことでソートを行うアルゴリズムであるマージソートについて解説する。同アルゴリズムの特徴と抽象化された手順を説明した後に、Pythonによる実装とその解説によりアルゴリズムの動作を理解させる。演習として、同アルゴリズムの動作過程を出力するPythonプログラムを実装させる。
		12_3	5キーワード	①ソートアルゴリズム　②バブルソート　③選択ソート　④挿入ソート　⑤マージソート
		12_4	コマ要素	□実務連携　□理解度確認テスト　■オリジナル教材　■ICT活用　■実習・実技・実験・演習　□該当なし
13	アプリケーション開発（1）	13_1	コマ主題細目	①コンソールアプリケーションの説明　②コンソールアプリケーションの実装　③コンソールアプリケーションの追加実装
		13_2	細目レベル	①コンソール上で動くアプリケーション実装の例としてゲームの開発を行う。まず、コンソールアプリケーションについて説明し、開発を行う上で最低限必要になるプログラムの雛形について解説する。雛形として、while文を用いて無限ループを発生させ、input関数、break文などを用いて、特定の文字列を入力したら無限ループを終了するといった簡素なプログラムを用意する。 ②①で用意した雛形を使い、簡単なゲームの開発の演習を行う。 ゲームの内容は、フィールド上に駒を配置することで、同フィールド上に置かれた敵オブジェクトを取り除くといったものである。 ゲームの仕様について解説し、雛形を元に制御構文や関数、コレクションなど、これまで扱った構文のうちクラスを除いたものを使って実装させる。 ③②で実装したゲームに、追加の仕様を実装し、より複雑なゲームの開発を行う。 追加のゲームの仕様は、敵オブジェクトや駒をランダムで配置し、駒を動かして敵オブジェクトを取り除いていくという内容である。駒は1つだけ配置するものとする。 ゲームの追加の仕様について解説し、②で開発したゲームを元に制御構文や関数、コレクションなど、これまで扱った構文のうちクラスを除いたものを使って実装させる。
		13_3	5キーワード	①コンソールアプリケーション　②while文　③input関数　④関数　⑤コレクション
		13_4	コマ要素	□実務連携　□理解度確認テスト　■オリジナル教材　■ICT活用　■実習・実技・実験・演習　□該当なし

14	アプリケーション開発(2)	14_1	コマ主題細目	①オブジェクト指向によらないコンソールアプリケーションの実装とその問題点　②オブジェクト指向によるコンソールアプリケーションの実装
		14_2	細目レベル	①オブジェクト指向プログラミングの利点を認識させるため、まずクラスやオブジェクト指向プログラミングを用いずに、前回講義で作成したプログラムに追加の実装を行う。追加実装の内容としては、フィールドに配置する駒や敵の数を増やすといったものである。前回講義で扱った、制御構文や関数、コレクションのみを用いて実装を行わせる。 そのうえで、この方法で実装したプログラムの問題点としてプログラムの保守性が低下することなどを挙げ、オブジェクト指向プログラミングを用いない実装の限界を理解させる。 ②①で挙げた問題点が、オブジェクト指向プログラミングを用いることで解決されることを簡単に説明した後に、実装例の説明を行う。駒や敵、フィールドをクラスとして定義し、インスタンス化することで複数の駒や敵を実現する例を示して解説することで、クラス、インスタンス、オブジェクト指向プログラミングなどの概念への理解を深める。加えて、クラスやメソッドの定義方法、__init__メソッドや__str__メソッドなど、以前の講義でも扱った内容を、実際のプログラムで用いる例を解説することで、実践的な理解を促す。 そのうえで、作成したプログラムを参照しながら、①で挙げた問題点が実際に解決されていることを説明し、オブジェクト指向プログラミングの利点を踏まえた活用方法を理解させる。加えて、オブジェクト指向プログラミングに基づいてさらに追加の仕様を実装させる演習を行う。
		14_3	5キーワード	①コンソールアプリケーション　②オブジェクト指向プログラミング　③クラス　④インスタンス　⑤オブジェクト指向プログラミングの利点
		14_4	コマ要素	□実務連携　□理解度確認テスト　■オリジナル教材　■ICT活用　■実習・実技・実験・演習　□該当なし
15	アプリケーション開発(3)	15_1	コマ主題細目	①リファクタリング　②継承を使ったコンソールアプリケーションの実装
		15_2	細目レベル	①リファクタリングとは、外から見たプログラムの振る舞いを変えずに、容易な理解や修正を実現するようにプログラムの内部構造を改善することである。 本コマでは追加の仕様を実装した後に、リファクタリングによりプログラムを改善していくため、まずリファクタリングの概要について実例を交えて説明する。リファクタリング前後のプログラムを比較して、保守性が向上していることを示すことで、リファクタリングの利点を理解させる。 ②継承を使ったコンソールアプリケーション開発の演習を行う。 前回講義で作成したプログラムを元に、追加の仕様を実現したゲームを実装させる。実装には、オブジェクト指向プログラミングにおける継承を用いる。 追加の仕様として、異なる動きをする複数の種類の駒を実現する。まず、継承を用いない愚直な実装を紹介して、その後、プログラムをリファクタリングすることで、継承関係を用いたプログラムに書き換える方法を説明する。最終的には、各駒の共通するメソッドを持たせるような親クラスを定義し、異なる動きを実現するようなメソッドを持つ子クラスを定義するような実装方法を説明する。 さらに、継承を用いたプログラムにリファクタリングする前と後のプログラムを比較することで、オブジェクト指向プログラミングにおける継承を活用することで、プログラムの重複を取り除くことができ、プログラムの変更容易性や理解容易性を改善できることを理解させる。
		15_3	5キーワード	①オブジェクト指向プログラミング　②リファクタリング　③継承　④親クラス　⑤子クラス
		15_4	コマ要素	□実務連携　□理解度確認テスト　■オリジナル教材　■ICT活用　■実習・実技・実験・演習　□該当なし