2026 年度シラバス

科目分類/Subject Categories			
学部等/Faculty	/工芸科学部 : /School of Science and Technology	今年度開講/Availability	/有 : /Available
学域等/Field	/設計工学域 : /Academic Field of Engineering Design	年次/Year	/3年次 : /3rd Year
課程等/Program	/情報工学課程・課程専門科目 : /Specialized Subjects for Undergraduate Program of Information Science	学期/Semester	/後学期 : /Second term
分類/Category	/ : /	曜日時限/Day & Period	/月 3 : /Mon.3

科目情報/Course Information				
時間割番号 /Timetable Number	12222201			
科目番号 /Course Number	12260020			
単位数/Credits	2			
授業形態 /Course Type	講義 : Lecture			
クラス/Class				
授業科目名 /Course Title	プログラミング言語論 : Programming Languages			
担当教員名 / Instructor(s)	/澁谷　雄 : SHIBUYA Yu			
その他/Other	インターンシップ実施科目 Internship	国際科学技術コース提供科目 IGP	PBL 実施科目 Project Based Learning	DX 活用科目 ICT Usage in Learning
				○
	実務経験のある教員による科目 Practical Teacher			
科目ナンバリング /Numbering Code				

授業の目的・概要 Objectives and Outline of the Course	
日	言語はものの考え方の枠組みを決める。それはプログラミング言語においても同じである。本講義では、まず C 言語等の一般的な命令型言語の要素がどのようにソフトウェア設計及びプログラミングと関連しているかを理解する。また、命令型言語以外にもいろいろなプログラミング言語があることを学び、それらの言語の特徴を理解する。特に関数型言語については、演習を交えて理解する。
英	We think everything using our language such as Japanese or English.　In programming, it is true.　This lecture presents the elements of imperative languages such as C language and their relationship to programming and software design. Moreover, other types of languages, that is, functional languages and logic languages, and some concepts, that is, object orientation and parallel/concurrent processing, are introduced. Especially, functional languages are introduced with some practice.

学習の到達目標 Learning Objectives	
日	プログラミング言語の役割を理解する。 命令型言語の基礎概念を理解し、ソフトウェア設計開発法との関連を説明できる。 オブジェクト指向概念の特徴を説明できる。 関数型言語の基礎概念を理解し、その構成要素と特徴を説明できる。 論理型言語の基礎概念を理解し、その構成要素と特徴を説明できる。 並行処理の基礎概念を理解し、その性質を説明できる。
英	Able to explain the role of programming languages. Able to explain the relationship of the basic concepts and features of imperative languages and the software

	design/development methods. Able to explain the features of object oriented concepts. Able to explain the elements and features of functional programming languages. Able to explain the elements and features of logic programming languages. Able to explain the properties of concurrent/parallel processing.

学習目標の達成度の評価基準 / Fulfillment of Course Goals（JABEE 関連科目のみ）	
日	
英	

授業計画項目 Course Plan			
No.		項目 Topics	内容 Content
1	日	プログラミング言語とは	高水準プログラミング言語の役割と特徴。アセンブリ言語との比較。
	英	What is the programming languages?	the role and features of high level programming languages
2	日	ソフトウェア設計法とプログラミング言語(1)	プログラミングの難しさ、プログラミングスタイル、それらに関連する命令型プログラミング言語の機能。
	英	Software development methods and imperative programming languages (1)	the difficulty of programming, programming style, and the functionalities of imperative languages related to them
3	日	ソフトウェア設計法とプログラミング言語(2)	構造化プログラミング、段階的詳細化と、それらに関連する命令型プログラミング言語の機能。
	英	Software development methods and imperative programming languages (2)	structured programming, stepwise refinement, and the functionalities of imperative languages related to them
4	日	ソフトウェア設計法とプログラミング言語(3)	手続きと関数、引数、スコープルール、モジュール化と、それらに関連する命令型プログラミング言語の機能。
	英	Software development methods and imperative programming languages (3)	procedure/function, parameter passing, scope rules, modular programming and the functionalities of imperative languages related to them
5	日	抽象データ型とオブジェクト指向	型理論、処理とデータ、情報隠蔽、モジュール化プログラミング。 クラスとオブジェクト、継承、多態。オブジェクト指向分析／設計とオブジェクト指向プログラミング。UML。
	英	Abstract data types and object orientation	Abstract data types and object orientation
6	日	オブジェクト指向言語の例:C++	オブジェクト指向言語の活用。
	英	Object oriented programming language: C++	practical use of object oriented programming language
7	日	関数型プログラミング言語(1)	関数型プログラミング言語の基礎。
	英	Functional programming language (1)	fundamental knowledge of functional programming language
8	日	関数型プログラミング言語(2)	関数型プログラミング言語の特徴。
	英	Functional programming language (2)	characteristic of functional programming language
9	日	関数型プログラミング言語(3)	関数型プログラミング言語の活用。 高階関数。データの抽象化と関数の抽象化。

	英	Functional programming language (3)	practical use of functional programming language, higher order functions, data abstraction, and function abstraction
10	日	論理型言語(1)	関係、述語、単一化。
	英	Logic programming languages (1)	relation, predicate, and unification
11	日	論理型言語(2)	Prolog。
	英	Logic programming languages (2)	Prolog
12	日	プログラミング作法	名前、式と文、一貫性と慣用句、関数マクロ、マジックナンバー、コメント。
	英	Practice of programming	variable name, expression, consistency, macro, magic number, comments
13	日	並行処理	プロセス間の関係。共有メモリ、メッセージパッシング、同期、クリティカルセクションと相互排除。
	英	Concurrent/parallel processing	the relationship between the processes, shared memory, message passing, synchronization, critical section, and mutual exclusion
14	日	様々なプログラミング言語	
	英	Other programming languages	Overview of several kinds of programming languages
15	日	まとめ	本講義のまとめ。
	英	Summary	summary

履修条件 Prerequisite(s)	
日	「プログラミングⅠ」、「プログラミングⅡ」、「データ構造とアルゴリズム」を履修していること。
英	Courses students need to have taken courses "Programming I," "Programming II," and "Data structures and algorithms" before.

授業時間外学習（予習・復習等） Required study time, Preparation and review	
日	毎回復習のために 1 時間程度の学習時間が必要である。また、課題レポートが課されたときにはさらに 1 時間程度の学習時間が必要となる。その他，最終レポート提出のために学習時間が必要である.
英	About 1 hour is needed for each report required. About 1 hour is needed for reviewing of each week. Some hours for learning are needed before final assignment.

教科書／参考書 Textbooks/Reference Books	
日	教科書なし。PDF ファイルを配布する。
英	No text book.　PDF files will be distributed.

成績評価の方法及び基準 Grading Policy	
日	学期末試験 80%、講義途中での数回の課題レポート 20%により評価する。合計点が 60 点以上を合格とする。
英	The evaluation of this course will be conducted based on the evaluation of final exam.(80%) and the evaluation of the required reports (20%). The students whose result of the evaluation is 60 points or higher will pass.

留意事項等 Point to consider	
日	・端末持参の有無やその利用内容は授業で指示します。 ・レポートは、文章を引用する際は、引用箇所が明確にわかるようにし、出典を記載すること。度を超えた引用は慎むこと。引用部分は誤字を含めて改変しないこと。

	・他人が作成したレポートを自分が作成したとして提出しないこと。
英	* Whether you need to bring your own device and how to use it will be specified during the lessons. * When quoting text in reports, ensure that the quoted section is clearly identifiable and cite the source. Avoid excessive quoting. Do not alter quoted sections, including any typographical errors. * Do not submit reports created by others as your own work.