Course Specifications

Course Title:	Compiler Design
Course Code:	CSI413
Program:	Computer Science and Information Technology
Department:	Computer Science and Information.
College:	College of Science in Zulfi
Institution:	Majmaah University

Table of Contents

A. Course Identification

1. Credit hours:3

2. Course type
a. University ☐   College ☐   Department √   Others ☐
b.   Required √   Elective ☐

3. Level/year at which this course is offered: 7th

4. Pre-requisites for this course (if any)**:**
CSI 222
CSI 221

5. Co-requisites for this course (if any)**:**
None

6. Mode of Instruction (mark all that apply)

No	Mode of Instruction	Contact Hours	Percentage
1	Traditional classroom	42	70%
2	Blended	6	10%
3	E-learning	6	10%
4	Correspondence	0	0%
5	Other	6	10%

7. Actual Learning Hours (based on academic semester)

No	Activity	Learning Hours
Contact Hours		
1	Lecture	30
2	Laboratory/Studio	20
3	Tutorial	10
4	Others (specify)	
	Total	60
Other Learning Hours*		
1	Study	10
2	Assignments	3
3	Library	2
4	Projects/Research Essays/Theses	1
5	Others (specify)	4
	Total	20

* The length of time that a learner takes to complete learning activities that lead to achievement of course learning outcomes, such as study time, homework assignments, projects, preparing presentations, library times

B. Course Objectives and Learning Outcomes

1. Course Description

The goal of this course is to introduce the design and implementation of compilers. Topics include: compiler organization, algorithms for lexical, syntactic and semantic analysis, top-down and bottom-up parsing, symbol table organization, error detection and recovery, intermediate and object code generation, and code optimization. Student has to implement a compiler for a simple high level language (like mini C) as a project.

2. Course Main Objective

The goal of this course is to introduce the design and implementation of compilers. Topics include: compiler organization, algorithms for lexical, syntactic and semantic analysis, top-down and bottom-up parsing (e.g., recursive descent, LL, LR, LALR parsing), symbol table organization, error detection and recovery, intermediate and object code generation, and code optimization. Student has to implement a compiler for a simple high level language (like mini C) as a project .

3. Course Learning Outcomes

	CLOs	Aligned PLOs
1	**Knowledge:**	
1.1	Understand the structure of compilers	k1
1.2	Understand the basic techniques used in compiler construction such as lexical analysis,	k1
1.3	top-down, bottom-up parsing, context-sensitive analysis, and intermediate code generation.	k1
2	**Skills :**	
2.1	Explain the core issues of Compiler design	S1
2.2	Design and implement a compiler using a software engineering approach	S1
2.3	Identify problems, and explain, analyze, and evaluate various design strategies of compilers.	S1
3	**Competence:**	
3.1	Work in a group and learn time management.	C3
3.2	Learn how to search for information through library and internet.	C3
3.3	Communicate with teacher, ask questions, solve problems, and use computers.	C3
3.4	Use Information technology and computer skills to gather information about a selected topic	C3

C. Course Content

No	List of Topics	Contact Hours
1	**Introduction to Compilers:** The role of language translation in the programming process, Comparison of interpreters and compilers, Language translation phases, Machine dependent and machine independent aspects of translation, Language translation as a software engineering activity	8
2	**Lexical Analysis:**	8

	Application of regular expressions in Lexical Analysis, Scanning, hand coded scanner vs. automatically generated scanners, formal definition of tokens, implementation of finite state automata.	
3	**Syntax Analysis and Parsing:** Revision of formal definition of grammars, BNF and EBNF, Bottom-up vs. Top-down parsing, Tabular vs. Recursive-descent parsers, Error handling,	12
4	**Parser Generators:** Automatic generation of tabular parsers, Symbol table management, Use of tools in support of the translation process	12
5	**Semantic Analysis:** Data type as set of values with set of operations, data types, Type-checking models, Semantic models of User defined types, Parametric polymorphism, Subtype polymorphism, Type checking algorithms.	8
6	**Intermediate Code Generation:** Intermediate and object code, intermediate representations, implementation of code generators	12
Total		60

D. Teaching and Assessment

1. Alignment of Course Learning Outcomes with Teaching Strategies and Assessment Methods

Code	Course Learning Outcomes	Teaching Strategies	Assessment Methods
1.0	**Knowledge**		
1.1	Understand the structure of compilers	Lectures Lab presentations	Mid-terms exams Lab exam Homeworks Final exams
2.0	**Skills**		
2.1	Explain the core issues of Compiler design	Lectures Lab Individual presentations	Mid-terms exams Lab exam Homeworks Discussions Final exams
2.2	Design and implement a compiler using a software engineering approach		
2.3	Identify problems, and explain, analyze, and evaluate various design strategies of compilers.		
3.0	**Competence**		
3.1	Work in a group and learn time management.	Small group discussion Whole group discussion Brainstorming Presentation demonstrations Case studies	Discussions Presentations
3.2	Learn how to search for information through library and internet.		
3.3	Communicate with teacher, ask questions, solve problems, and use computers.		
3.4	Use Information technology and computer skills to gather information about a selected topic		

2. Assessment Tasks for Students

#	Assessment task*	Week Due	Percentage of Total Assessment Score
1	First written mid-term exam	6	15%
2	Second written mid-term exam	12	15%
2	Presentation, class activities, and group discussion	Every week	10%
3	Homework assignments	After every chapter	10%
4	Practical exam	15	10%
5	Final written exam	16	40%

Assessment task (i.e., written test, oral test, oral presentation, group project, essay, etc.)

E. Student Academic Counseling and Support

Arrangements for availability of faculty and teaching staff for individual student consultations and academic advice :

1. 6-office hours per week in the lecturer schedule.
2. The contact with students by e-mail , mobile, office telephone and website.

F. Learning Resources and Facilities

1.Learning Resources

Required Textbooks	Dick Grune, Kees van Reeuwijk, Henri E. Bal, Ceriel J.H. Jacobs, Koen Langendoen, "Modern Compiler Design", ISBN 978-1-4614-4698-9, Springer New York Heidelberg Dordrecht London, 2nd 2012.
Essential References Materials	**https://www.isi.edu/~pedro/Teaching/CSCI565-Spring16/**
Electronic Materials	**https://www.coursera.org/**
Other Learning Materials	Video and presentations that are available with the instructor.

2. Facilities Required

Item	Resources
Accommodation (Classrooms, laboratories, demonstration rooms/labs, etc.)	☐ Class Rooms ☐ Computer Labs ☐ Library
Technology Resources (AV, data show, Smart Board, software, etc.)	Smart Board, C++
Other Resources	None

Item	Resources
(Specify, e.g. if specific laboratory equipment is required, list requirements or attach a list)	

G. Course Quality Evaluation

Evaluation Areas/Issues	Evaluators	Evaluation Methods
Analysis of students' results.	Teaching Staff	Direct
Observation during work.	Teaching Staff	Indirect
Students' evaluations.	Teaching Staff	Indirect
Colleagues' evaluations.	Peer Reviewer	Direct

Evaluation areas (e.g., Effectiveness of teaching and assessment, Extent of achievement of course learning outcomes, Quality of learning resources, etc.)
Evaluators (Students, Faculty, Program Leaders, Peer Reviewer, Others (specify)
Assessment Methods (Direct, Indirect)

H. Specification Approval Data

Council / Committee	DEPARTMENT COUNCIL
Reference No.	
Date	