Pokhara University
Faculty of Science and Technology

Course No.: xxx xxx	Full marks: 100
Course title: Programming in C (3-1-3)	Pass marks: 45
Nature of the course: Theory & Practical	Time per period: 1 hour
Level: Undergraduate	Total Periods: 45
	Program: BE

1. Course Description

This course first introduces the different computer/programming languages- low-level and high-level languages and generation of the programming languages. This course enables the students to solve a given problem using a structured programming language- the C language through the problem-solving steps- problem analysis, design of algorithm, flowcharting, coding the algorithm, executing and compiling/interpreting the program, testing and debugging the program and finally well documenting the program for future understanding. The aim of this course is to provide the basic knowledge of structured programming concepts to develop a program using the C language.

2. General Objectives

The general objectives of this course are:

- To provide the basic knowledge of computer/programming languages and generations of programming languages.
- To provide the basic knowledge of problem-solving steps to resolve a particular problem using a computer program.
- To provide the basic concepts of structured programming with the C language.

3. Methods of Instruction

3.1.General instructional Techniques: Lecture, discussion, readings.
3.2.Specific instructional Techniques: Lab works, Project works

4. Contents in Detail

Specific Objectives	Contents
• familiar with the various types of programming languages, programming paradigms and software • analyze a problem, find the solution, implement the solution using a particular computer/programming language and finally prepare its documentation	**Unit I: Programming Languages and Problem Solving (6 hrs)** 1.1 Programming Languages- machine-level language, assembly language and high-level language. 1.2 Software and its types 1.3 Generations of programming languages 1.4 Structured programming 1.5 Problem solving using computer- Problems Analysis (understanding of the problem, feasibility and requirement analysis), Design (Algorithm and flowchart), Coding (compilation/interpretation and execution),

	Testing and debugging, Implementation, Evaluation and Maintenance of computer programs, Program documentation
• explain the history and features of C programming, implement the various constructs of C programming like constants, variables, keywords, data types, operators and formatted I/O functions etc. • write and evaluate the expressions in C programs	**Unit II: Introduction to C (6 hrs)** 2.1 History of C language 2.2 Features of C 2.3 The C as a middle-level language 2.4 The C as a system programming language 2.5 The C character set 2.6 Keywords and Identifiers 2.7 Data types 2.8 Constants, variables and their declaration 2.9 Formatted input/output functions 2.10 The C Operators
• implement the various control structures of C programming with reference to algorithm or flowchart they developed already	**Unit III: Control Structures (6 hrs)** 3.1 Introduction and types of control statements- sequential, branching and looping statements 3.2 Branching statements- simple if statement, if-else, nested if, if-else-if ladder and switch statements 3.3 Looping statements- for loop, while loop, do-while loop, nested loop 3.4 The break, continue and goto statements
• represent/store homogenous type of data as array and realize the multi-dimensional array representation • implement string handling functions to handle textual data	**Unit IV: Arrays and Strings (6 hrs)** 4.1 Introduction to arrays 4.2 One dimensional and Multidimensional arrays 4.3 Initialization of arrays and accessing the elements of arrays 4.4 Strings- the character arrays 4.5 Functions related to the strings
• use function to perform certain operation • use macro definition, header files and function prototype and implement them in their program	**Unit V: Functions (6 hrs)** 5.1 Introduction 5.2 Importance of functions 5.3 Returning a value from a function and sending a value to a function 5.4 Function prototypes 5.5 Calling a function- Call by value 5.6 Recursive functions 5.7 Passing an array to a function 5.8 Local variables, formal parameters and global variables 5.9 Storage classes 5.10 Pre-processor directives- C libraries, macros and header files
• implement the pointers in program	**Unit VI: Pointers (6 hrs)**

• allocate and deallocate memory dynamically in C language	6.1 Introduction 6.2 Pointer operators 6.3 Pointer arithmetic 6.4 Returning multiple values form functions using pointers 6.5 Pointers and Arrays 6.6 Double indirection 6.7 Dynamic memory allocation
• able to understand how the heterogenous data are represent and stored in structure and union and implement them in program	**Unit VII: Structure and Union (5 hrs)** 7.1 Definition of Structure 7.2 Nested-Structure 7.3 Array of Structure 7.4 Structures and Pointers 7.5 Union 7.6 Self-referential structure
• create files, access the contents of the file and perform the required operations in the file.	**Unit VIII: Files and File Handling (4 hrs)** 8.1 FILE pointer, File opening modes (read, write, append) 8.2 File handling functions 8.3 Creating and operating a file in different modes

5. **List of Experiments**

Laboratory works should cover all the concepts of C language studied in the lectures. Students should submit a final project that uses all the constructs and features of C studied in this course. The marks for the practical work will be based on the project work.
Tools: any C/C++ compiler

6. List of Tutorials

The various tutorial activities that suit this course should cover all the content of this course to give students a space to engage more actively with the course content in the presence of instructor. Students should submit tutorials as assignments or class works to the instructor for evaluation. The following tutorial activities of 15 hrs should be conducted to cover all the content of this course:

A. Discussion-based Tutorials: (2 hrs)
1. Evolution of Programming languages and its generations (Class discussion)
2. Software and its types.
3. Generations of programming languages.
4. Structured programming. (Oral Presentation).

B. Problem solving-based Tutorials: (10 hrs)
5. Develop algorithms and flowcharts to solve various problems such as to find largest number among three numbers, prime numbers, temperature conversion, product of matrices, finding sum of the terms in series, printing various patterns etc.
6. Develop the C programs for the problems for which you developed the algorithms.
7. Write a program to pass an array to a function.
8. Write a program to use pointers to pass multiple values from a function.

9. Write a program to use the basic string functions to manipulate string data.
10. Write a program to use the principle of recursion to solve the complex problems such as to find factorial of a number, fibonacci series.
11. Write a program to illustrate the macros and header files.
12. Write a program to to illustrate how memory is allocated and deallocated in C language.
13. Write a program to use the nested structure. Discuss the scenarios when the structures and unions are used in real practice.
14. Write a program to solve simple file handing problems.

C. Review and Question/Answer-based Tutorials: (3 hrs)

15. Case study of "Development of C with the UNIX operating system and origin of C++ languages" followed by Oral Presentation in class.
16. Students ask questions within from the course content and assignments and review key course content in preparation for tests or exams.

7. **Evaluation System and Students' Responsibilities**

Evaluation System

The internal evaluation of a student may consist of assignments, attendance, test-exams, term-exams, lab reports and projects etc. The tabular presentation of the internal evaluation is as follows:

External Evaluation	**Marks**	**Internal Evaluation**	**Weight**	**Marks**
Semester-End examination	50	**Theory**		30
		Attendance & Class Participation	10%	
		Assignments	20%	
		Presentations/Quizzes	10%	
		Term exam	60%	
		Practical		20
		Attendance & Class Participation	10%	
		Project Report	10%	
		Viva	20%	
		Exam	60%	
		Total Internal		50
Full Marks: 50 + 50 = 100				

Student Responsibilities

Each student must secure at least 45% marks in internal evaluation with 80% attendance in the class in order to appear in the Semester End Examination. Failing to get such score will be given NOT QUALIFIED (NQ) and the student will not be eligible to appear the Semester-End Examinations. Students are advised to attend all the classes, formal exam, test, etc. and complete all the assignments within the specified time period. Students are required to complete all the requirements defined for the completion of the course.

8. Prescribed Books and References

Text Books

1. Gottfried, B. *Programming with C.* New Delhi: McGraw-Hill.

2. Balagurusamy, E. *C programming.* New Delhi: McGraw-Hill.

References

1. Kelley A. & Pohl, I. *A Book on C, Programming in C,* USA: Addison-Wesley.
2. Kernighan B. W. & Ritchie, D. M. *The C Programming Language.* USA: Prentice Hall Publication.