



B24 Computers and Programming

3:0

Program Outcomes addressed

- a. An ability to apply knowledge of engineering, information technology, mathematics, and science
- c. An ability to design a system or component, or process to meet stated specifications

Competencies

1. Select computers for different applications.
2. Comprehend the nature of problems that a computer can solve extremely well - be able to list 5 non-trivial, interesting problems (unique in their own way) which are difficult to solve for a human being but can be solved easily by a computer.
3. Comprehend the following terms in the context of problem solving by a computer: Problem specification, input-output analysis, algorithm, flowchart, pseudo-program, programming language, assembly language, machine language, compiler, assembler, program correctness.
4. Explain the difference between arrays and linked lists, and create two examples where arrays are better than linked lists and two examples where linked lists are better than arrays.
5. Explain the difference between iteration and recursion, and create two examples where iteration is better than recursion and two examples where recursion is better than iteration.
6. Design the flowchart and write efficient code for problems like
 - Recursive and iterative programs for binary search
 - Recursive and iterative programs for Fibonacci numbers
 - Recursive and iterative programs for finding the GCD of two numbers
 - Reverse a linked list while traversing it only once
7. Explain the role of pointers in implementing singly linked lists, doubly linked lists, binary trees, and general trees.
8. Explain the reason why different constructs are available for iteration, such as "for" loops, "do...while" loops.

Assessment Pattern

	Bloom's Category	Test 1	Test 2	End-semester examination
1	Remember	20	10	0
2	Understand	20	20	10
3	Apply	50	40	50
4	Analyze	10	20	20



5	Evaluate	0	10	20
6	Create	0	0	0

Course Level Learning Objectives:

Remember

1. What is a Computer?
2. Name the different I/O devices used with a computer?
3. What is the difference between system software and application software?
4. List five programming languages commonly used/
5. What is the role of operating system in a computer?
6. What is structured programming?
7. What are the various data types?
8. What are the I/O functions in C?
9. What are format specifiers?
10. What is a header file?
11. What are the various looping statements?
12. Give the Syntax of switch-case statement?
13. What is an array?
14. What is a pointer?
15. What is the significance of function?

Understand:

1. Differentiate translators with Compilers?
2. Differentiate Loader with a Linker?
3. Compare while loop with do – while Loop?
4. What are the advantages of using Macro?
5. Explain how recursive functions affect the run time efficiency?
6. Differentiate between Structure and Union in C.
7. Explain how dynamic arrays are efficient compared to Static with example?
8. How is memory managed in C?
9. What are the advantages of using Command line Arguments?
10. How garbage collection is done in C?

Apply

1. Write a Macro to find the Armstrong number between 1 and 1000?
2. Write a recursive function to calculate the Combinatory of a nCr?
3. Write a program to perform stack operation using pointers?
4. Write a program to perform linked list operation using pointers?



5. Write a program to generate the pay slip of an employee using dangling if – else statement?
6. Write a program to compute Matrix Multiplication using Pointers?
7. Write a program to read the content of the file and copy it to another file?
8. Write a program to check if the given word is available in the file or not?

Analysis

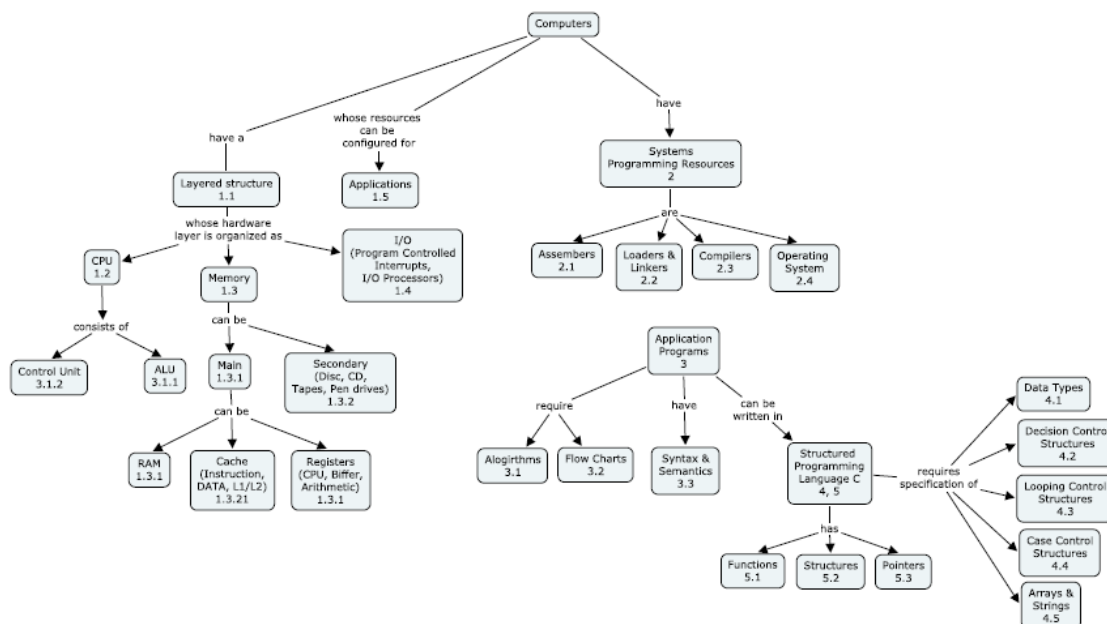
1. Explain the difference between the following:
 - (i) Program testing and debugging.
 - (ii) Top down and bottom up approaches.
 - (iii) Interpreted and compiled languages
2. Why are pointers so powerful? Analyze their efficiency giving an example?
3. Are the following statements valid? Justify your answer
 - (i) $k = (\text{char}^*)\& m$
 - (ii) $m = (\text{float}^*)\& p$
4. Is there any advantage of using recursion over looping control structures? Give a suitable example?
5. Analyze the factors that influence the execution times of a program?
6. Represent the functional blocks of a 1 pass assembler as a diagram?
7. Outline the functions of the various loader schemes?
8. Illustrate the **Limitation of array of pointers to strings** using a sample?
9. Differentiate the keywords **BREAK** and **CONTINUE**?

Evaluate

1. Execution time affects efficiency more than storage space. Justify?
2. Compare and contrast Structures with union?
3. Justify the need for **Type Casting** over **Type Conversion**?
4. Compare and contrast **IO mapped IO** with **Memory mapped IO**?
5. Summarize the various built in **String** functions?
6. Evaluate the sorting procedure using arrays and pointers?
7. Given an Educational institution try to automate it acquiring the needed resources?



Concept Map:



Lecture Schedule

No.	Topic	No. of Lectures
1	Introduction to computers	
1.1	Layered Structure of a computer	1
1.2	CPU	2
1.3	Memory	2
1.4	Input/Output	2
1.5	Configuring resources of computers for applications	1
2	Systems Programming Computers	
2.1	Assemblers	3
2.2	Loaders and Linkers	4
2.3	Compilers	2
2.4	Operating Systems	2
3	Application Programming	
3.1	Algorithms	1



3.2	Flowcharts	2
3.3	Syntax, semantics and execution	2
4.	Structured Programming Language	
4.1	Symbols and data types	1
4.2	Looping control structures	1
4.3	Decision control structures	2
4.4	Case control structures	2
4.5	Arrays and Strings	2
5.	Functions and Pointers	
5.1	Functions	3
5.2	Structures	2
5.3	Pointers	2

Syllabus

Introduction to computers: Layered Structure of a computer, CPU, Memory, Input/Output, Configuring resources of computers for applications **Systems Programming Computers:** Assemblers, Loaders and Linkers, Compilers, Operating Systems **Application Programming:** Algorithms, Flowcharts, Syntax, semantics and execution, **Structured Programming Language:** Symbols and data types, Looping control structures, Decision control structures, Case control structures, Arrays and Strings, **Functions and Pointers:** Functions, Structures, and Pointers

References

1. Leland L. Beck: System Software, Pearson Education, 3rd Edition, 2004
2. John. J Donovan: System Programming, Tata McGraw Hill Edition, 2000
3. Yashavant Kanetkar: Programming in ANSI C, 2nd Edition-BPB Publications
4. Yashavant Kanetkar: Let us C, BPB Publications 8th Edition 2007
5. Yeshavant Kanetkar: Understanding Pointers in C, 2nd Edition BPB Publications

Course Designers:

1. Askarunisha aacse@tce.edu
2. M. Vijayalakshmi mviji@tce.edu
3. S. Prasanna sprcse@tce.edu