R23ES07 JNTUGV -R23
I Year-I Semester
INTRODUCTION TO PROGRAMMING
(Common to All branches of Engineering)

UNIT-I INTRODUCTION TO COMPUTER PROBLEM SOLVING

SHORT ANSWER QUESTIONS

1. Define Computer

A computer is an electronic device, operating under the control of instructions stored in its own memory that can accept data (input), process the data according to specified rules, produce information (output), and store the information for future use. A computer is a system made of two major components: hardware and software. The computer hardware is the physical equipment. The software is the collection of programs (instructions) that allow the hardware to do its job.

2. Define Program

The set of explicit and unambiguous instructions expressed in a programming language to solve a problem is called a program. A program is as an algorithm expressed in a programming language.

3. What is an Algorithm?

An algorithm consists of a set of explicit and unambiguous finite steps which, when carried out for a given set of initial conditions, produce the corresponding output and terminate in a finite time.An algorithm is a corresponding solution to a problem that is independent of any programming language.

4. What is Divide-and-Conquer strategy?

The basic idea with divide-and-conquer is to divide the original problem into two or more subproblems which can hopefully be solved more efficiently by the same technique. If it is possible to proceed with this splitting into smaller and smaller subproblems we will eventually reach the stage where the subproblems are small enough to be solved without further splitting.

5. Define Dynamic Programming

This method is used most often when we have to build up a solution to a problem via a sequence of intermediate steps. This method relies on the idea that a good solution to a large problem can sometimes be built up from good or optimal solutions to smaller problems.

6. Draw the steps of problem solving approach

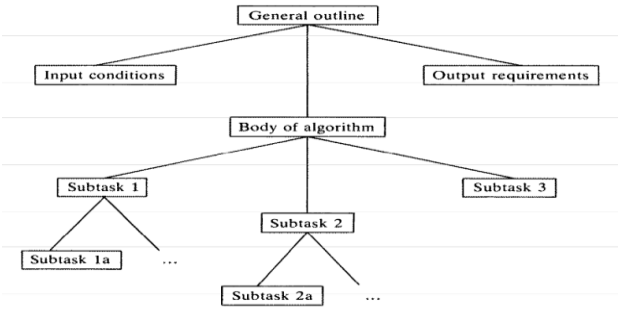

Fig. 1.1 Schematic breakdown of a problem into subtasks as employed in top-down design.

7. What is top-down approach ?

Top-down design is a strategy that we can apply to take the solution of a computer problem from a vague outline to a precisely defined algorithm and program implementation.

8. List the steps followed by top-down design

1. Breaking a problem into subproblems
2. Choice of a suitable data structure
3. Construction of loops
4. Establishing initial conditions for loops
5. Finding the iterative construct
6. Termination of loops

9. Write the steps to design an algorithm

1. Use of procedures to emphasize modularity
2. Choice of variable names
3. Documentation of programs
4. Debugging programs
5. Program testing

10. Define Program verification

Program verification refers to the application of mathematical proof techniques to establish that the results obtained by the execution of a program with arbitrary inputs are in accord with formally defined output specifications.

11. What is Computation state?

The progress of a computation from specific input conditions through to termination can be thought of as a sequence of transitions from one computation state to another. Each state, including the initial state, is defined by the values of all variables at the corresponding point in time.

12. How to make state transition?

A state transition and progress towards completion is made by changing the value of a variable followed by a transfer of control to the next instruction on the current execution path.

13. Define verification condition (VC)

Symbolic execution enables us to transform the verification procedure into proving that the input assertion with symbolic values substituted for all input variables implies the output assertion with final symbolic values substituted for ail variables. A proposition phrased in this way is referred to as a verification condition (VC) over the program segment from the input assertion to the output assertion.

14. Define loop invariant

A loop invariant should be a property (predicate) that captures the progressive computational role of the loop while at the same time remaining true before and after each loop traversal irrespective of how many times the loop is executed. Once the loop invariant is established, there are several steps that must be taken to verify the loop segment.

15. Give suggestions to improving efficiency

1. Redundant computations
2. Referencing array elements
3. Inefficiency due to late termination
4. Early detection of desired output conditions
5. Trading storage for efficiency gains

16. Define order notation

A standard notation has been developed to represent functions which bound the computing time for algorithms. It is an order notation and it is usually referred to as the O-notation. An algorithm in which the dominant mechanism is executed cn2 times for ¢, a constant, and n the problem size, is said to have an order n2 complexity which is written as O(n2). Formally, a function g(n) is O(f(n)) provided there is a constant ¢ for which the relationship

$g(n) \leq cf(n)$

holds for all values of n that are finite and positive.

LONG ANSWER QUESTIONS

1. What are computer problem solving requirements? Explain
2. Explain about improving efficiency of an algorithm
3. Describe algorithm analysis in detail
4. Explain about phases of problem solving
5. Explain the steps involved in top down approach
6. Describe program verification in detail