Algorithms – Theory and practice questions

What is an algorithm ?

A computer algorithm is a detailed set of instructions, when executed sequentially, solves a problem

Steps to be followed to develop an algorithm :

1. Understand the problem
2. Identify the output of the problem
3. Identify the inputs required by the problem to reach the desired output
4. Design a logic to produce the desired result
5. Test the algorithm
6. Repeat step 1 to 5 till desired results are produced

Algorithm for Making a Mobile call

1. Pick up the Mobile.
2. Open mobile MENU
3. Select Contacts option.
4. Search appropriate Person to whom you want to call.
5. Select the searched contact.
6. Go to Option.
7. Press call button

While writing algorithms we will use following symbol for different operations:

‘+’ for Addition

‘-’ for Subtraction

‘*’ for Multiplication

‘/’ for Division and

‘←’ for assignment. For example A← X*3 means A will have a value of X*3

Two forms of Algorithm:

- Pseudo code
- Flow chart

Advantages of Pseudo code

- Reduced complexity.
- Increased flexibility.
- Ease of understanding.

Problem 1: Find the area of a Circle of radius r.

Inputs to the algorithm:

Radius r of the Circle.

Expected output:

Area of the Circle

Algorithm:

```
Step1: Read\input the Radius r of the Circle
Step2: Area ← PI*r*r // calculation of area
Step3: Print Area
```

Problem2: Write an algorithm to read two numbers and find their sum.

Inputs to the algorithm:

First num1.

Second num2.

Expected output:
Sum of the two numbers.

Algorithm:

```
Step1: Start
Step2: Read\input the first num1.
Step3: Read\input the second num2.
Step4: Sum ← num1+num2 // calculation of sum
Step5: Print Sum
Step6: End
```

Problem 3: Convert temperature Fahrenheit to Celsius
Inputs to the algorithm:
Temperature in Fahrenheit
Expected output:
Temperature in Celsius
Algorithm:

```
Step1: Start
Step 2: Read Temperature in Fahrenheit F
Step 3: C ← 5/9*(F32)
Step 4: Print Temperature in Celsius: C
Step5: End
```

Type of Algorithms
The algorithm and flowchart, classification to the three types of control structures. They are:
1. Sequence
2. Branching (Selection)
3. Loop (Repetition)

The branch refers to a binary decision based on some condition. If the condition is true, one of the two branches is explored; if the condition is false, the other alternative is taken. This is usually represented by the ‘if-then’ construct in pseudo-codes and programs. In flowcharts, this is represented by the diamond-shaped decision box. This structure is also known as the selection structure.

Problem4: write algorithm to find the greater number between two numbers

```
Step1: Start
Step2: Read/input A and B
Step3: If A greater than B then C=A
Step4: if B greater than A then C=B
Step5: Print C
Step6: End
```

Problem5: write algorithm to find the result of equation:
f(x) = { -x, if x < 0 and x if x >=0

```
Step1: Start
Step2: Read/input x
Step3: If X Less than zero then F=-X
Step4: if X greater than or equal zero then F=X
Step5: Print F
Step6: End
```

Problem6: A algorithm to find the largest value of any three numbers.

```
Step1: Start
Step2: Read/input A,B and C
Step3: If (A>=B) and (A>=C) then Max=A
Step4: If (B>=A) and (B>=C) then Max=B
```

```
Step5:If (C>=A) and (C>=B) then Max=C
Step6: Print Max
Step7: End
```

The loop allows a statement or a sequence of statements to be repeatedly executed based on some loop condition. It is represented by the ‘while’ and ‘for’ constructs in most programming languages,

Problem7: An algorithm to calculate even numbers between 0 and 99

```
1. Start
2. I ← 0
3. Write I in standard output
4. I ← I+2
5. If (I <=98) then go to line 3
6. End
```

Problem8: Design an algorithm which gets a natural value, n,as its input and calculates odd numbers equal or less than n. Then write them in the standard output:

```
1. Start
2. Read n
3. I ← 1
4. Write I
5. I ← I + 2
6. If ( I <= n) then go to line 4
7. End
```

Problem9: Design an algorithm which generates even numbers between 1000 and 2000 and then prints them in the standard output. It should also print total sum:

```
1. Start
2. I ← 1000 and S ← 0
3. Write I
4. S ← S + I
5. I ← I + 2
6. If (I <= 2000) then go to line 3
else go to line 7
7. Write S
8. End
```

Problem10: Design an algorithm with a natural number, n, as its input which calculates the following formula and writes the result in the standard output:

$S = \frac{1}{2} + \frac{1}{4} + ... + 1/n$

```
1. Start
2. Read n
3. I ← 2 and S ← 0
4. S= S + 1/I
5. I ← I + 2
6. If (I <= n) then go to line 4
else write S in standard output
7. End
```

Problem 11: Algorithm for product of first n natural numbers

```
1. Start
2. Set Product equal to 1
3. For i = 1 to n do
4. Product = Product * i
5. Output Product
6. Stop
```

Problem 12: Algorithm for Greatest of Three Numbers

1. Start
2. Read A, B, C
3. If A > B is True, then check whether A > C, if yes then A is greatest otherwise C is greatest
4. If A > B is False, then check whether B > C, if yes then B is greatest otherwise C is greatest
5. Stop

Problem 13: Algorithm to find whether a number is odd or even

1. Start
2. Input number n
3. Compute num = n%2
4. Check value of num, if num =0, then Print “Even” else Print “Odd”
5. Stop

Problem 14: Algorithm for area and circumference of circle

1. Start
2. Input r
3. Area = π * r* r
4. Circumference = 2 * π *r
5. Output Area, Circumference
6. Stop

Flowchart

- **A flowchart can therefore be used to:**

– Define and analyze processes
– Build a step-by-step picture of the process for analysis, discussion, or communication
– Define, standardize or find areas for improvement in a process

Summary / Important definitions

Algorithm is the sequence of steps to be performed in order to solve a problem by the computer.

Three reasons for using algorithms are efficiency, abstraction and re-usability.

Algorithms can be expressed in many different notations, including natural languages, pseudo code, flowcharts and programming languages.

Analysis of algorithms is the theoretical study of computer program performance and resource usage, and is often practiced abstractly without the use of specific programming language or implementation.

The practical goal of algorithm analysis is to predict the performance of different algorithms in order to guide program design decisions.

Most algorithms do not perform the same in all cases; normally an algorithm’s performance varies with the data passed to it. Typically, three cases are recognized: **the best case, average**

case and worst case.

Worst case analysis of algorithms is considered to be crucial to applications such as games, finance and robotics.
O-notation, also known as Big O-notation, is the most common notation used to express an algorithm's performance in a formal manner.

Flowchart is a graphical or symbolic representation of an algorithm. It is the diagrammatic representation of the step-by-step solution to a given problem.

Flowcharts are used in analyzing, designing, documenting or managing a process or program in various fields.

Benefits of using flowcharts include ease of communication, effective and efficient analysis and coding, proper documentation and maintenance.
Limitations of using flowcharts include **complex logic and multiple modifications.**

The types of flowcharts are High-Level Flowchart and Detailed Flowchart.

Program Design consists of the steps a programmer should do before they start coding the program in a specific language