**Assignment Title:** Revised Questions part I
**Course Title:**
**Course Code: CSC 206**

**Question**
kindly use this questions to revised the topics we have treated so far. It will come in batches; this is the first batch The rest will be posted as times goes on. kindly attempt it and submit online. CSC 206: COMPUTER PROGRAMMING II REVISED QUESTIONS 2019/2020

1. Define the following: I. Programming ii. A program iii. Programming languages
2. There are three types of programming languages, describe them.
3. There are some conventional features which a programming language must possess, what are they?
4. Write short note on six programming methodologies.
5. You are given a task as an IT consultant to write a proposal on five major stages that are involved in developing an efficient program for Johnson LTD.
6. As the Head of programming development team of Otito IT Solutions, kindly advise other team members’ considerations in writing good programs.
7. What is structured programming?
8. \*Highlight and discuss fundamental objectives of structured programming.
9. With a given example, briefly write short note on Logical pattern that characterize Structured Programming.
10. \*As a computer analyst, you are given a task to solve a Time Table scheduling problem in College of Science, Afe Babalola University. What is the method you are to employ in solving the task and what are the necessary properties required of the steps to take to resolve the issue?
11. Write an algorithm using a flowchart to find the volume of a sphere, given the radius as 23.
12. Write an algorithm using Pseudocode to find the radius of a cone, if the volume is given as 200cm3 and height as 18.
13. Differentiates between object oriented languages and structured oriented languages.
14. // program to find the circumference of a circle. #include #define pi 3.142 int main ()

{const float rad = 7.5; float cir; cir = 2 \* pi \* rad; printf(“the value of circumference of a circle is %d”, cir); return 0; } Describe each line in the above program.

**Answers**

1. Programming is the process of writing programs.

ii. A program is a set of codes that instructs the computer to carry out some processes.

iii. Programming languages are language through which we can instruct the computer to carry out some process or tasks.

1. Machine language: also known as machine code, is the native language the native language that the CPU understands, because it only uses a binary system, an element of notations containing only a series of numbers consisting of one and zero.

b. Assembly Language: it is the language that consists of symbolic codes, which are easier to remember than machine codes.

c. High-level Language: they are user-friendly languages which are similar to English with the vocabulary of words and symbols. They are easier to learn and require less time to write.

1. **Syntax and Structure**: Programming languages for commands can overlap just like when using words in spoken languages. To produce text to screen in Ruby or [Python](https://www.python.org/about/gettingstarted/), you are to use ‘print command’ similar to using imprimer and imprimir when we want to print in French and Spanish.

**Functionality of Languages:** All these languages can make the same functionality, similar to how all spoken languages can reflect the same phrases, objects, and emotions.

**Natural Lifespan:** Programming languages are created when a talented programmer attempts to create a fresh way or an easier method of expressing a computational idea. He presents this idea to his fellow programmers for approval. If the other programmers come to an agreement, they implement the language and use it for their programs. Hence, the programming language spreads and becomes existent.

**Written in English:** As opposed to spoken languages (except English), nearly all languages are written in English. This is true whether one is programming JavaScript, Ruby, HTML, Brazilian, Ruby, or Python. They all use English syntax codes and keywords even if the programmers who create them are French or Chinese.

**One Programmer:** As opposed to spoken languages, programming languages can be made by only one programmer or a single creator. Languages that can be created by only one creator include JavaScript (Brendan Eich), Ruby (Yukihiro Matsumoto), and Python (Guido van Rossum).

1. Procedural Programming: it is a series of steps, each of which performs a calculation, retrieves input or produces output.

ii. Concurrent Programming: it is collection of cooperating processes, sharing information with each other from time to time, but generally asynchronous.

iii. Logic (Declarative) Programming: it is a collection of logical declarations about what outcome a function should accomplish rather than how the outcome should be accomplished.

iv. Functional Programming: it is a collection of mathematical functions, each with an input and a result. Interaction and combination of functions are carried out by functional compositions, conditionals and recursion.

v. Event Driven Programming: In this type of programming paradigm, flow of execution is determined by the events like user clicks or other programming threads or query result from database.

### vi.Object-oriented Programming: Here the solution revolves around entities or objects that are part of problem. The solution deals with how to store data related to the entities, how the entities behave and how they interact with each other to give a cohesive solution.

### Defining the Problem: You meet with users from Johnson LTD to analyze the problem, or you meet with a systems analyst who outlines the project. Specifically, the task of defining the problem consists of identifying what it is you know, and what it is you want to obtain. Eventually, you produce a written agreement that, among other things, specifies the kind of input, processing, and output required.

### ii. Planning the Solution: Two common ways of planning the solution to a problem are to draw a flowchart and to write pseudocode, or possibly both.

### iii. Coding the Program: As the programmer, your next step is to code the program that is, to express your solution in a programming language. You will translate the logic from the flowchart or pseudocode or some other tool to a programming language. Then your coded program must be keyed, probably using a terminal or personal computer, in a form the computer can understand.

### iv. Testing and Debugging: the process involves the location and removal of error in the program if any. Testing is the process of checking if the program is working as expected and finding errors in the program. Debugging is the process of correcting errors that are found (An error in a program is called a bug).

### v. Documenting the Program: This is the final stage of program Development. Typical program documentation materials include the origin and nature of the problem, a brief narrative description of the program, logic tools such as flowcharts and pseudocode, data-record descriptions, program listings, and testing results. The wise programmer continues to document the program throughout its design, development, and testing.

### Naming Conventions: it is very important to give meaningful names to all your constructs. A name like get Height () or get\_avg\_height() gives us much more information than ctunde(). Also, a variable name total for addition is more meaningful than pen. The name of the class should communicate its purpose.

### File Naming and Organization: Files should be organized into directories in a module-wise fashion instead of having a monolithic structure where all source code files and all header files are in a single directory. This should be a part of the design process.

### Formatting and Indentation: The lines within the code should be clearly organized in a way that it will be easy to read and understand even for the writer. Proper identification should be used to show subordinate lines.

### Comments and Documentation: Introducing comments and proper explanations (documentation) of the program aid in understanding the code. They help us in following the program flow, and skip parts for which we are not interested in details. This allows for program amendment and extensibility.

### Classes: Ensure that all classes in your application have a default constructor, copy constructor and overloaded operator.

### Functions: A function should normally do only one job and do it well. Avoid generic functions with lots of conditional branches to do everything. If a function is supposed to do multiple jobs, then create helper functions and delegate responsibilities to them. Make functions simple and small. The ideal size of functions is around 35-40 lines.

### Using STL: Use Standard Template Library (STL) instead of creating your own container data structures.

### Structured Programming is a programming paradigm aimed at improving the clarity, quality and development time of computer programs by making extensive use of subroutines, block structures, FOR and WHILE loops.

### 9i. Sequence: it refers to an ordered execution of statements. It has one entry and exit point.

### ii. Selection: it uses conditions and one of a number of statements is executed depending on the state of the program. Examples if, then, else, end if, switch or case.

### iii. Repetition: a statement is executed until the program reaches a certain state, or operations have been applied to every element of a collection. Examples; while, repeat, for or do…. until.

### 11.

Print v

V=4/3\*22/7\*23\*23\*23

Read r

 start

end

1. #include<stdio.h>

#include<conio.h>

void main()

{

    int r;

    float pi=3.14,area,ci;

    clrscr();

    printf("Enter the Radius of a Circle:");

    scanf("%d",&r);

    area=pi\*r\*r;

    printf("Area of a Circle is: %f\n",area);

    ci=2\*pi\*r;

    printf("Circumference is: %f",ci);

    getch();

}