Computer and Programming Aptitude (General Q&A)

1. **Difference between Data and information.**
   Definition1- Data are simply facts or figures — bits of information, but not information itself. When data are processed, interpreted, organized, structured or presented so as to make them meaningful or useful, they are called information. Information provides context for data.
   Definition2- Data and information are interrelated. Data usually refers to raw data, or unprocessed data. It is the basic form of data, data that hasn't been analyzed or processed in any manner. Once the data is analyzed, it is considered as information. Information is "knowledge communicated or received concerning a particular fact or circumstance." Information is a sequence of symbols that can be interpreted as a message. It provides knowledge or insight about a certain matter.

2. **What is an instruction in computer?**
   An **instruction** is a segment of code that contains steps that need to be executed by the computer processor.
   Or an instruction is an order given to a computer processor by a computer program.

3. **What is a Command in computer?**
   A **command** is an instruction telling a **computer** to do something, such a run a single program or a group of linked programs.

4. **What is a Process in computer?**
   A **process** is an instance of a program running in a **computer**. A process is started when a program is initiated

5. **What is a Task in computer?**
   Whenever you execute a program, the operating system creates a new task for it. The task is like an envelope for the program: it identifies the program with a *task number* and attaches other book keeping information to it.

6. **Why C is called high level language?**
   C is called a high level language because it enables a programmer to write programs that are more or less independent of a particular type of computer. Such languages are considered high-level because they are closer to human languages and further from machine languages.
   In contrast, assembly languages are considered low-level because they are very close to machine languages.

7. **What is high level language in Computer Programming?**
   High-level languages are designed to be used by the human operator or the programmer. They are referred to as "closer to humans." In other words, their programming style and context is easier to learn and implement than low-level languages, and the entire code generally focuses on the specific program to be created.

8. **What is a Source code and Object code?**
   The source code consists of the programming statements that are created by a programmer with a text editor or a visual programming tool and then saved in a file. For example, a programmer using the C language types in a desired sequence of C language statements using a text editor and then saves them as a named file. This file is said to contain the source code. It is now ready to be compiled with a C compiler and the resulting output, the compiled file, is often referred to as object code. The object code file contains a sequence of instructions that the processor can understand but that is difficult for a human to read or modify.

9. **What is a compiler?**
   A compiler converts a high level language program/code into binary instructions (machine language) that our computer can interpret, understand and take the appropriate steps to execute the same.

10. **What is difference between compiler and interpreter?**
    Interpreter: analyzes and executes each line of source code in succession, without looking at the entire program. The advantage of interpreters is that they can execute a program immediately.

Compilers: require some time before an executable program emerges. However, programs produced by compilers run much faster than the same programs executed by an interpreter.

11. What are the steps involved in C Compilation?

1. **Preprocessing** This is the first phase through which source code is passed. This phase include:
    - Removal of Comments
    - Expansion of Macros
    - Expansion of the included files
2. **Compilation** is the second pass. It takes the output of the preprocessor, and the source code, and generates assembler source code.
3. **Assembly** is the third stage of compilation. It takes the assembly source code and produces an assembly listing with offsets. The assembler output is stored in an object file.
4. **Linking** is the final stage of compilation. It takes one or more object files or libraries as input and combines them to produce a single (usually executable) file. In doing so, it resolves references to external symbols, assigns final addresses to procedures/functions and variables, and revises code and data to reflect new addresses (a process called relocation).

12. What is the difference between compilation and linking?

Compilation: A compiler takes your file containing source code and translates it to machine code. It then places that machine code into an output file called an object file and has the file extension .obj.
Each of the items your files provides or needs has a name, a symbol. The object file also has lists of those symbols that your code provides and of those symbols which it needs. The process of translating the source code into an object file is called compiling.
Linking: After the compiler has created all the object files, another program is called to bundle them into an executable program file. That program is called a linker and the process of bundling them into the executable is called linking.

13. What is Machine code?

Machine code, also known as machine language, is the elemental language of computers, comprising a long sequence of binary digital zeros and ones (bits).

14. Why C is called machine independent language?

A **machine-dependent language** works only on a specific computer system and its components. What it means is a code that you write on one particular system may not run in other system with different configurations.
A **machine—independent language** is a computer language that works on different computer systems regardless of their components. Thus a program written in one machine would run on any other machine.
For example: A program written on a machine with "xyz" configurations would work perfectly fine when run on a machine with "abc" configurations.
Note**: C language is a machine independent language because no matter which machine one writes in, it can be run on any other machine.**

15. Why C is called structured language?

C is called a structured programming language because to solve a large problem, C programming language divides the problem into smaller modules called functions or procedures each of which handles a particular responsibility.
Advantages:

- C structured programming is simple and easy to understand and implement.
- It is well suited for small size implementation. However this is not restricted. A good design can extend it to large size implementation.
- Programmers do not require to know complex design concepts to start a new program.

16. Why C is called procedural language?

The word "procedure" is the key element here to notice. It means "***a set of procedures***" which is a "***set of subroutines***" or a "***set of functions***". We all know about "functions in C language". **'C'** is a procedure oriented language. In a POP (**Procedure Oriented Programming)** method, emphasis is given to functions or subroutines. Functions are a set of instructions which performs a particular task. Functions are called repeatedly in a program to

execute tasks performed by them. For example, a program may involve collecting data from user (reading), performing some kind of calculations on the collected data (calculation), and finally displaying the result to the user when requested (printing). All the 3 tasks of reading, calculating and printing can be written in a program with the help of 3 different functions which performs these 3 different tasks.

17. What is difference between System software and Application software?

Subject	Application Software	System Software
Definition	Application software is computer software designed to help the user to perform specific tasks.	System software is computer software designed to operate the computer hardware and to provide a platform for running application software.
Purpose	It is specific purpose software.	It is general-purpose software.
Classification	• Package Program, • Customized Program	• Time Sharing, • Resource Sharing, • Client Server • Batch Processing Operating System • Real time Operating System • Multi-processing Operating System • Multi-programming Operating System • Distributed Operating System
Environment	Application Software performs in a environment which created by System/Operating System	System Software Create his own environment to run itself and run other application.
Execution Time	It executes as and when required.	It executes all the time in computer.
Essentiality	Application is not essential for a computer.	System software is essential for a computer
Number	The number of application software is much more than system software.	The number of system software is less than application software.

18. What is a variable?

A variable is known as a named memory location.

A **variable** is an entity that is used to store data. Without variables, there is no way (or actually NO PLACE) to store data. A variable has

- a name (more specifically a symbolic name)
- an associated physical memory space (portion in a RAM)
- a data type
- a value (depends on data type)
- a scope
- a lifetime

19. What are identifiers?

Identifiers are sequences of characters used for naming variables, functions, new data types, and preprocessor macros.

RULES:

1. Identifier can be include letters uppercase (A-Z) and lowercase(a-z)
2. Identifier can be decimal digits (0-9). The first character of an identifier cannot be a digit.
3. You Can Include underscore character _ in identifiers.
4. The first character of an identifier cannot be a underscore.

5. Lowercase letters and uppercase letters are distinct, such that foo and FOO are two different identifiers.

20. Explain constant in C.

A constant is an entity whose value does not change. A constant can either be a numeric constant or a literal constant.
Example of numeric constants:

```
1
500
-10000.100
3.1416
```

Example of literal constants:

```
'A'
"Hello"
"The quick brown fox jumped over the lazy dog."
```

Note: In C, a literal constant with only one character is referred to simply as a **character**. It is written such that it is enclosed in a pair of single quotes. If there is more than one character, it is called a **string**. A string is written enclosed in a pair of double quotes.

21. What is a Keyword/Reserved words?

Keywords are predefined; reserved words used in programming that have special meanings to the compiler. Each keyword is meant to perform a specific function in a C program .Keywords are part of the syntax and they cannot be used as an identifier. There are 32 keywords in C language.
For example: int money;
Here, *int* is a keyword that indicates *'money'* is a variable of type integer.

22. Explain C tokens.

Each and every smallest individual unit in a C program is known as C token.
C tokens are of six types. They are,

1. Keywords (eg: int, while),
2. Identifiers (eg: main, total),
3. Constants (eg: 10, 20),
4. Strings (eg: “total”, “hello”),
5. Special symbols (eg: (), {}),
6. Operators (eg: +, /,-,*)

23. What are Types of operators (unary, binary, ternary)?

The symbols which are used to perform logical and mathematical operations in a C program are called **C operators.** These C operators join individual constants and variables to form expressions.
Operators, functions, constants and variables are combined together to form **expressions.**
Consider the expression A + B * 5. where, +, * are operators, A, B are variables, 5 is constant and A + B * 5 is an expression.

1. **Unary Operator:** Unary operators are those operators which need only one operand to perform an operation. These operators are usually written on left of variables or objects but sometimes can be written on right as well. For example !x, ++x .
   Types of unary operators are:
   Increment operator: **“ ++ ”,** Decrement operator: **“ - - “**, Negation operator: **“ ! ”**, Unary minus operator: **“ – “**
2. **Binary Operator:** A binary operator is an operator that operates on two operands and manipulates them to return a result. Operators are represented by special characters or by keywords and provide an easy way to compare numerical values or character strings.
   Binary operators are presented in the form: Operand1 Operator Operand2
   Some common binary operators in computing include: Equal (==)
   Not equal (!=)
   Less than (<)
   Greater than (>)

Greater than or equal to (>=)
Less than or equal to (<=)
Logical AND (&&)
Logical OR (||)
Plus (+)
Minus (-)
Multiplication (*) , Divide (/)

3. **Ternary operator**: A Ternary operator is an operator that operates on two operands. The first argument is a comparison argument, the second is the result upon a true comparison, and the third is the result upon a false comparison. Ternary operator has the following form**: exp1 ? exp2 : exp3**.
   An expression **a ? b: c** evaluates to b if the value of a is true, and otherwise to **c** .

24. What is garbage value in C?

1. Allocating a variable implies reserving some memory for that variable. In some programming languages (like **C**) if a variable is allocated but not assigned, it is said to have a "**garbage value**".
2. When you define a variable, like: **int** x;
a small block of memory is allocated to the variable. However, we have only declared the variable, and not initialized it, which means that the block of memory that has been allocated to the variable still contains some value that has been left over from previous programs and operations. That value is called a garbage value. This may lead to erroneous results in programs.
To avoid this, declare and initialize variables like this:
int x = 0;

25. What is type conversion/type casting in C?

Type cast is basically a conversion from one type to another. It helps us to compute expressions containing variables of different data types.

There are two types of type conversion:

1. **Implicit Type Conversion** Also known as 'automatic type conversion'.
   a. Done by the compiler on its own, without any external trigger from the user.
   b. Generally takes place when in an expression more than one data type is present. In such condition type conversion (type promotion) takes place to avoid lose of data.

Example of Type Implicit Conversion:

```
// An example of implicit conversion
#include<stdio.h>
int main()
{
   int x = 10;    // integer x
   char y = 'a';  // character c

   // y implicitly converted to int. ASCII
   // value of 'a' is 97
   x = x + y;

   // x is implicitly converted to float
   float z = x + 1.0;

   printf("x = %d, z = %f", x, z);
   return 0;
}
```

Output:

```
x = 107, z = 108.000000
```

2. Explicit Type Conversion– This process is also called type casting and it is user defined. Here the user can type cast the result to make it of a particular data type.

The syntax in C:

```
(type) expression
```

Type indicated the data type to which the final result is converted.

```
// C program to demonstrate explicit type casting
#include<stdio.h>

int main()
{
   double x = 1.2;
   // Explicit conversion from double to int
   int sum = (int)x + 1;
   printf("sum = %d", sum);
   return 0;
}
```

Output: sum = 2

26. What is the meaning of precedence of operator?

Operator precedence describes the order in which C reads expressions. The operator precedence chart contains the answers. Operators higher in the chart have a higher precedence, meaning that the C compiler evaluates them first. Operators on the same line in the chart have the same precedence, and the "Associativity" column on the right gives their evaluation order.

27. What is a data type in programming language?

A data type specifies

- the kind of values that can be assumed by a variable of that type
- The range of values that can be assumed by a variable of that type
- the amount of memory (in bytes) needed by a variable to store a value of that type

28. What are data types in C?

There are four basic data types, namely:

- char - the character data type. The char data type is used to represent/store/manipulate character data values. A char data type value requires one byte of memory space. The range of values that can be assumed by a char value is from 0 to 255. The number-to-character coding that is used is ASCII.
- int - the integer data type. The int data type is used to represent/store/manipulate signed whole numeric values.
- float - the single precision floating point data type. The float data type is used to store single precision signed real numbers(i.e., those numbers with decimal values).
- double - the double precision floating point data type. The double data type is used to store double precision signed real numbers(i.e., bigger numbers with decimal values).

The amount of memory space required to store an int, a float and double is **platform-dependent** (depends on the machine and the software). For 32-bit machines (and 32-bit compilers such as Microsoft C compiler), an int and float requires 4 bytes of memory each, while a double requires 8 bytes of memory.

Note that a char data is actually numeric (0 to 255), and is treated as a subset of int values. Any operation (discussed later) on integer values can also be performed on characters.

How can you determine the actual size of each data type?

The following program can be used to print the sizes of the basic data types:

```
#include <stdio.h>

int main(int argc, char *argv[])
{
```

```
    printf("Size of char: %d\n", sizeof(char));
    printf("Size of int: %d\n", sizeof(int));
    printf("Size of float: %d\n", sizeof(float));
    printf("Size of double: %d\n", sizeof(double));
    _getch();
    return 0;
}
```

The `sizeof` is a keyword in C. It is an operator that yields the size in number of bytes of the specified data type shown above.

29. What is an algorithm?

An algorithm is a finite sequence of instructions, a logic and explicit step-by-step procedure for solving a problem starting from a known beginning.

The sequence of steps to be performed in order to solve a problem by the computer is known as an algorithm.

In mathematics, computer science, and related subjects, an algorithm is a finite sequence of steps expressed for solving a problem.

An algorithm can be defined as "a process that performs some sequence of operations in order to solve a given problem". Algorithms are used for calculation, data processing, and many other fields.

30. What is a Flowchart?

A flow chart is a graphical or symbolic representation of a process. Each step in the process is represented by a different symbol and contains a short description of the process step. The flow chart symbols are linked together with arrows showing the process flow direction.

31. What is difference between Compile time Error and Run time Error?

1. Compile time error: You get this error when you compile the program. With compilation error you will not be able to run or execute the program.

2. Run time error: You can compile the program successfully. When you run the program, you will get error or exception.

Run Time Error

C runtime errors are those errors that occur during the execution of a c program and generally occur due to some illegal operation performed in the program.

Examples of some illegal operations that may produce runtime errors are:

- Dividing a number by zero
- Trying to open a file which is not created
- Lack of free memory space

Compile Errors

Compile errors are those errors that occur at the time of compilation of the program.

C compile errors may be further classified as:

1. Syntax Errors

A ***syntax error*** is an ***error*** in the source code of a program. Since computer programs must follow strict ***syntax*** to compile correctly, any aspects of the code that do not confirm to the ***syntax*** of the ***programming language*** will produce a ***syntax error***.

When the rules of the c programming language are not followed, the compiler will show syntax errors.

For example, consider the statement,

int a,b:

The above statement will produce syntax error as the statement is terminated with : rather than ;

2. Semantic Errors

Semantic errors are reported by the compiler when the statements written in the c program are not meaningful to the compiler.

For example, consider the statement,

1 b+c=a;

In the above statement we are trying to assign value of a in the value obtained by summation of b and c which has no meaning in c. The correct statement will be

```
1       a=b+c;
```

3. Logical Errors

Logical errors are the errors in the output of the program. The presence of logical errors leads to undesired or incorrect output and are caused due to error in the logic applied in the program to produce the desired output.

Also, logical errors could not be detected by the compiler, and thus, programmers has to check the entire coding of a c program line by line.

32. What is difference between entry control and exit control loop.

Entry Controlled Loop

Loop, where test condition is checked before entering the loop body, known as Entry Controlled Loop

Example: while loop, for loop

Exit Controlled Loop

Loop, where test condition is checked after executing the loop body, known as Exit Controlled Loop

Entry Controlled Loop	Exit Controlled Loop
Test condition is checked first, and then loop body will be executed.	Loop body will be executed first, and then condition is checked.
If Test condition is false, loop body will not be executed.	If Test condition is false, loop body will be executed once.
For loop and while loop are the examples of Entry Controlled Loop.	Do while loop is the example of Exit controlled loop.
Entry Controlled Loops are used when checking of test condition is mandatory before executing loop body.	Exit Controlled Loop is used when checking of test condition is mandatory after executing the loop body.

References:

https://en.wikipedia.org
https://www.quora.com/
https://www.computerhope.com ›
https://www.webopedia.com
techtarget.com/

Notes compiled by- Rajendra Singh Bisht