1Q,

A,

8 Bits in 1 Byte

1,048,576 Kilobytes make a gigabyte

2Q,

3Q,

Difference between compiled and interpreted languages

Compiled languages:

A compiled computer language outputs a program file that is directly understandable by the CPU in a computer and is ready to be run. It can be in binary format, is not readable by humans, but is directly understood by the computer.

Only translation occurs and executable file is created in compilation.

In a compiled implementation of a language, a compiler will translate the program directly into code that is specific to the target machine which is also known as machine code, basically code that is specific to a given processor and operating system then the computer will run the machine code on its own.

Examples of compiled languages C, C++, Java

Interpreted languages:

Whereas an interpreted computer language generally outputs a program in a manner that is not understood by the computer CPU. It must be turned into machine readable (binary format) when you run it. This file is not understood by the computer.

The file is Translated and executed at the same time.

In an interpreted implementation of a language, the file is compiled and executed line by line at run time, the source code is not directly run by the target machine what happens instead is that another program reads and then executes the original source code. This other program is also known as the interpreter. In interpretation, the original code is also typically converted into some intermediate code which is then processed by an interpreter that converts the code into machine specific code.

A good way to separate a compiled language from an interpreted language is to look at what happened to source code and how that source code is turned into a running program.

Examples of interpreted languages is Java.

JAVA is an interpreted language.

4Q,

Waterfall model: Define-before-design, design-before-code

When requirements are well known and product definition is stable using water fall method is a good practice.

It is easy to understand, widely used and known in theory.

Works well on stable products.

Projects where using waterfall will be best approach:

New version of an existing product.

Porting an existing product to a new platform.

Ex: Microsoft, what’s app,

Agile SDLC:

Agile is less formal and reduced scope,

Speed up and bypass one or more life cycle phases,

Used for time- critical applications

Continuous delivery of software,

Collaboration with customer,

Continuous update according to changes.

Projects where using waterfall will be best approach:

For new products.

5Q,

Key roles and responsibilities of QA Engineer in Software organization:

Reading various requirement documents to gain thorough and deep understanding of the project like business requirements document, software specification documents, and software design documents.

Coming up with test plan for the requirements.

Analyze test failures and report errors back to developers. Report bug and track them using bug tracking system. Follow up with developers until bug is fixed.

Monitor whole process and identify areas of efficiency improvements.

6Q,

Different sections of test plan for test cases and explain with examples

Test case id or name:

Naming the test cases, this should represent the functional area you are going to verify in that test case.

Ex: login pass test

Test case description:

All the details you are going to test, and the particular behavior being verified by the test.

“What I am going to test”

Enter as much as info as possible and test to be carried out, behavior being verified.

Ex: we want to check whether we are able to login

Preconditions:

Conditions to be met

Any assumptions that apply to a test along with any preconditions that must be met before test can be executed.

Ex: which page should user start the journey

Does this test depend on other test cases?

Input Test data:

Mention exact test data to be used for test

If the test needs some values to be verified give the value range

Ex: user details, password etc.

Test case steps:

Detail way how to go step by step by covering the functional flow

Ex: go to, enter details, hit enter etc.

Test result/ expected result:

Mention in detail what page / screen you expect to appear after the test.

Ex: whether log in, log out.

7Q

Object oriented programming:

Java is an object oriented programming, three main principles of object oriented program are encapsulation, polymorphism, and inheritance.

Object oriented programming is structured programming, in this you define the data and the routines that are permitted to act on that data. Thus, data type defines precisely what sort of operations can be applied to that data.

Polymorphism: “one interface, multiple methods”

This means that it possible to design a generic interface to a group of related activities.

Is the quality that allows one interface to access a general class of actions. The specific action is determined by the exact nature of the situation.

Ex: in java we create one general set of routines that works for all other specific situations.

Inheritance:

Is the process by which one object can acquire the properties of another object.

Using inheritance, an object need only define those qualities that makes it unique within its class. It can inherit its general attributes from its parent.

Ex: if you create class: ‘house’, object: ‘house 1’

Object ‘house 1’ inherits all the characteristics from its parent ‘house’.

8Q,