**Assignment 1**

1. What is JDK? JRE? JVM?
2. JDK is a kit that provides the environment to develop and execute the JAVA program.

It contains Development Tools (javac.exe and java.exe) and JRE

1. JRE is an installation package that provides an environment to only run (not develop) the java program.
2. JVM in Java is the engine that drives the **Java Code**. It converts Java bytecode into machines language and execute the java program.

1. What is JVM architecture

Diagram

Description automatically generated

1. Class Loader: prepares the Java classes and loads them into main memory.
2. Runtime Memory/Data Area: Holds the runtime variables and data.
3. Execute Engine: execute the Java program.
4. What is Class Loader architecture?

A picture containing diagram

Description automatically generated

Class loading subsystem has three phase:

1)Phase Load:

Loading is the process of finding the binary representation of a class or interface and creating a class or interface from that binary representation. The following image is the loading process.

Diagram

Description automatically generated

1. Bootstrap class loader is the parent of all the other Class Loader instances. It loads the packages like java. Lang and java. Utility
2. Extension class loader loads extension classes like ODBC and JDBC
3. Application class loader will load classes in your programs.

2)Phase link:

Phase link has three steps:

1. Verify: to check if it is a java binary code.
2. Prepare:

Memory is allocated for the static variables inside a class.

**IMP:  It is only memory allocation that is only for the class variables not for the instance variables. And this variable will be set to its default value like bool = false.**

**eg . private static integer.**

1. Resolve:

The Java virtual machine replaces class symbolic references with direct references.

Exception Class not found happens in this part.

3)Phase initialization

Initialize those static variables in class

1. What is Runtime area?

It has five areas:

1. PC register:

It is independent for each thread and contains the address of JVM instructions that the thread currently executing.

`2) Stack

It is local to each thread and stores parameters, local variables and return addresses during method calls.

3)Heap

It is shared by all threads and contains objects data. Objects are allocated here.

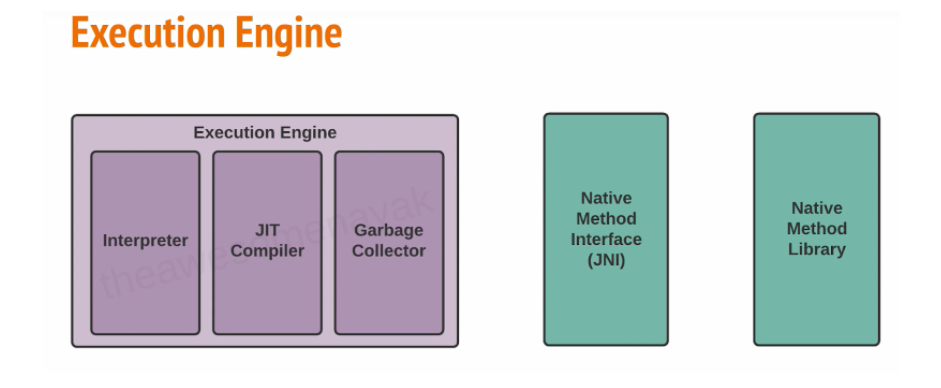
4)Method Area

It stores per-class structures such as the constant pool (more on this later), the static variables, the code (for constructors and methods, method data, etc).

5)Native method stack

It is used for those method which is non-Java.

1. What is execution engine?



Execution Engine contains following parts:

1)Interpreter: convert byte code into the machine code and execute them in a sequential Manner.

2)JIT compiler: help to improve the performance of Java program by compiling byte code into machine code at run-time.

3)Garbage Collector:

serial GC:

cmd in JVM:-XX:+UseSerialGC

parallel GC(default):

-XX:UseParallelGC

G1 GC(parallel):

divide heap into equal sized chunks and it prioritize different chunks according to which chunk has the most number of garbage.

-XX:++G1GC

GC process:

Diagram

Description automatically generated

Gc has three generation:

1. Young generation:
2. Old generation:
3. Permanent generation:
4. What is java compiler?

A Java compiler is a program that takes the text file work of a developer and compiles it into a platform-independent Java file. Java compilers include the Java Programming Language Compiler (javac), the GNU Compiler for Java (GCJ), the Eclipse Compiler for Java (ECJ) and Jikes.

1. Why is java platform independent?

Java is platform independent because the code you write runs inside of the Java Virtual Machine (JVM) instead of directly on the underlying platform.

1. What is IDE? Why is it important for developers?

Integrated Development Environment

IDEs increase programmer productivity by combining common activities of writing software into a single application: editing source code, building executables, and debugging.

1. Is java case sensitive?

Yes.

1. What do the following key words do?  
   static: Static methods/attributes can be accessed without creating an object of a class.

final: It makes classes ,attributes and methods impossible to inherit, override and change.

Table

Description automatically generated

Void: Specify that a method should not return a value.

Null:  **indicate that the variable does not refer to any object**

Package: Declares and create a package

Class: Define a class structure.

New: Creates new objects

1. What is primitive type and reference type?

Primitive types are the basic types of data: byte , short , int , long , float , double , boolean , char .

Reference type :  **Reference variables store addresses to locations in memory** for where the data is stored.

1. Is parameter passed by value or reference?

Java is always **pass-by-value**.

Java only has the two types of passing: by value for built-in types, and by value of the pointer for object types.

1. What is the output: System.out.println(1 > 0 : “A”:”B”);

A

1. How to define constants in java?

public class myClass {  
   static int days\_in\_week = 7;  
}

1. What is String? Is it primitive type?

Autoboxing is **the automatic conversion that the Java compiler makes between the primitive types and their corresponding object wrapper classes**

Strings in Java are **Objects that are backed internally by a char array.**

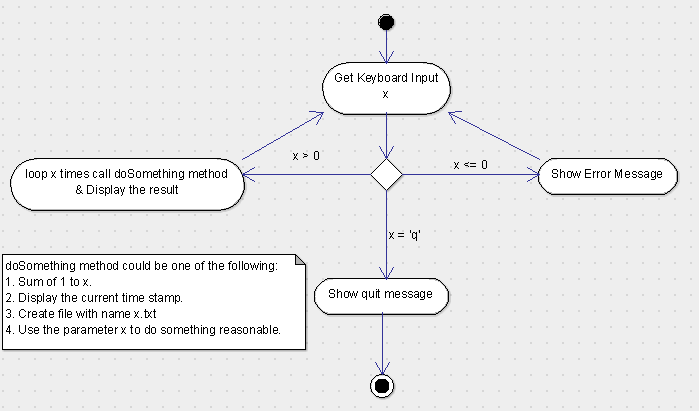
It is not a primitive type.

1. How to check if a String is representing a number?

Text

Description automatically generated

1. Write a program to implement the following activity diagram:



public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
 while (true)  
 {  
 String x = sc.nextLine();  
 if(x.equals("q"))  
 {  
 System.*out*.println("quit!");  
 break;  
 }  
 else  
 {  
 int num = x.charAt(0) - '0';  
 if(num > 0)  
 {  
 System.*out*.println(*doSomething*(num));  
 }  
 else  
 {  
 System.*out*.println("x<0 input error");  
 }  
 }  
 }  
  
}  
  
public static int doSomething(int x)  
{  
 int sum = 0;  
 for(int i = 1; i <= x; i++)  
 {  
 sum += i;  
 }  
 return sum;  
}

1. Write a program to merge two arrays of int.

public static int[] mergeIntArr(int[] arr1, int[] arr2)  
{  
 int[] rst = new int[arr1.length + arr2.length];  
 int x = 0;  
 int y = 0;  
 int c = 0;  
 while(x < arr1.length && y < arr2.length)  
 {  
 if(arr1[x] < arr2[y])  
 {  
 rst[c++] = arr1[x++];  
 }  
 else  
 {  
 rst[c++] = arr2[y++];  
 }  
 }  
 while(x<arr1.length)  
 {  
 rst[c++] = arr1[x++];  
 }  
 while(y<arr2.length)  
 {  
 rst[c++] = arr1[y++];  
 }  
 return rst;  
}

1. Write a program to find the second largest number inside an array of int.

public static int findTheSecondLarge(int[] arr)  
{  
 if(arr.length < 2)  
 {  
 return -1;  
 }  
 int maxNum = arr[0];  
 int secNum = 0;  
 for (int i = 1; i < arr.length; i++)  
 {  
 if(arr[i] > maxNum)  
 {  
 secNum = maxNum;  
 maxNum = arr[i];  
  
 }  
 else  
 {  
 if(arr[i] > secNum)  
 {  
 secNum = arr[i];  
 }  
 }  
 }  
 return secNum;  
}