OOP

* Constractor এর ভিতর কিছু (public/private হলে কি হয়?)

নাই

Head-first java

Interface:

Static:

what is Integer and int

Exception: //(custom exception)

Nested class:

Inner class:

throws:

throw

multithreading:

static দিয়ে ~~thea~~ thread synchronization কেমনে করে?

* Generics:
  * hash
  * hash set
  * hash tree

String handeling

Termtest: 1,2,7

Does java support multiple inheritance? If yes, how and if not, why?

⇒ Java doesn't support multiple inheritance because of two reasons -

1. In Java, every class is a child of object class. When it inherits from more than one supper class, sub class gets the ~~ob~~ ambiguity to acquire the property of object class

2. In Java every class has a constractor, if we write it explicitly or not at all. The first statement in calling super() to invoke the supper class constractor. If the class has more than one super class, it gets confused.

Can constractor be inherited?

⇒ No, Constractor can not be ~~in~~ inherited in java.

In case of inheritance, child/sub class inherit the state (data membe and ~~behaviour~~ behavior (methods) of parent/super class. But it does not inherits the constractor because of the following reason:

If parent class constractor is inherited in child class, then it can not be treated as constractor because constractor name must be same as class name. It will be treated as a

method but now the problem is, method should have explicit return type which inherited parent class constructor can not have.

※ Compare structure from c and class in java.

⇒

Basis for comparison	Structure	Class
Basic	If access specifier is not declared, by default all member are 'public'	If access specifier is not declared, by default all members are 'private'

Declaration	Struct structure_name{
	type struct_name1;
	type struct_name2;
	};
		class class_name{
		data member;
		member function;
		};

Instance	Instance of 'structure' is called 'structure variable'	Instance of a class is called 'object'.
Polymorphism and inheritance	Not supported	Supported polymorphism and a class can also be inherited.
Nature	Value type	Reference type
Memory is allocated on	Stack	Heap

Null valves	Not possible	can have null values

3) Difference between throw and throws -

throw	throwns
1. The throw keyword is used inside a function. It is used when it is required to throw an Exception logically.	1. The throwns keyword is used in the function signature. It is used when the function has some statements that can lead exceptions.
2. The throw keyword is used to throw an exception explicitly. It can throw only one exception at a time.	2. The throwns keyword can be used to declare multiple exceptions, separated by a comma. whichever exception occurs, if matches with the declaration one, is thrown automatically then.
3. Syntax wise throw keyword is followed by the instance variable.	3. Syntax wise throwns keyword is followed by exception class name

Q For Java, what does it mean by "write once, run anywhere"?

⇒ Java applications are called WORA (Write Once Run Anywhere). This means a programmer can develop Java code on one system and can expect it to run any other Java-enabled system without any adjustment. This is all possible because of JVM.

In traditional programming Language like C, C++ when program were compiled, they used to be converted into the code understood by the particular underlying hardware, which understands different code will cause an error, so you have to re-compile the code to be understood by the new hardware.

In Java, the program is not converted to code directly understood by Hardware; rather it is converted to "bytecode", which is interpreted by JVM, so once compiled it generates bytecode file, which can be run anywhere which has JVM and hence it gets the nature of write once & and run anywhere.

Q) How many ways we can create a thread? show example.

⇒ There are two ways we can create a thread.

① By extending thread class:

Steps:

1. Create a class and extends Thread
2. Override the run() method.
3. Instantiate the class.
4. Call start() method

Ex:

```
Public class ThreadExtends extends Thread {
  @override
    public void run(){
        Sout("Thread is running");
    }

    Public static void main(String[] Args){
      ThreadExtends myThread = new ThreadExtends();

       myThread.start();

      }

}
```

(ii) By implementing Runnable Interface

Steps:

1. Create a class and implement Runnable interface.
2. Implement the run() method.
3. Instantiate Runnable class.
4. Instantiate the Thread class, pass Runnable class in Thread's Constructor.
5. Call start() method.

Ex:

```
Public class ImplementRunnable implements Runnable{
  @override
  Public void run(){
    Sout("Thread is running");
  }
  Public static void main (String[] Args){
   ImplementRunnabe r=new ImplementRunnabe();
  Thread t = new Thread (r);
  t.start();
  }
}
```

B What is static 'static'? Why would you use them?

⇒ In Java programming language, the keyword static means that the particular member belong to a type itself, rather than to an instance of that type.

Static is a reserved keyword in Java. The main use of static is for memory management at JVM level. Static variable are stored in a special area called permanent generation

[The only drawback of static is that static variable memory is not garbage collected so we need to use it wisely.]

B Method overloading vs method overriding.

Overloading	Overriding
(i) Must have at least two methods by the same name in the class.	(i) Must have at least one method by the same name in both parent class and child class.
(ii) Must have different number of parameters.	(ii) Must have the same number of parameters.
(iii) If the number of parameters is the same, then it must have different types of parameters.	(iii) Must have the same parameter type.

(iv) Overloading is known as compile-time polymorphism	(iv) Overriding is known as runtime polymorphism.

18 Ques: Solving

A

1] (b) i) 1.

(vi) Can you declare constructor private? Explain.

⇒ yes, we can declare a constructor as private. If we declare a constructor private we are not able to create an object of a class. we can use this private constructor in the singleton Design pattern.

Condition for private constructor-

* A private constructor does not allow a class to be subclassed.
* A private constructor does not allow to create an object outside the class.
* If all the constant methods are there in our class we can use a private constructor.

* If all the methods are static than we can use a private constructor. (Because using those method, we don't need object of that cla

* If we try to extend a class which is having private constructor compile time error will occur.

(vi)

Q2)

(i) Why java is called platform independent? Explain using JVM and compare it to "c".

⇒ The meaning of platform-independent is that the java compiled code (byte code) can run on all operating system.

A program in written in a language that is human readable language. It may contain words, phrases, etc which the machine does not understand. For the source code to be understood by the machine, it needs to be in a language understood by machines, typically a machine-level language. So, a compiler converts high-level language into a format understood by the machines. This converted executable code can be executed by the cpu, or it may be an intermediate representation that is interpreted by a virtual machine. This intermediate representation in java is the java Byte code.

Step by step of a java program:-

⇒ whenever, a program in written in Java, the javac compiles it.

⇒ The javac result of the JAVA compiler is the .class file or the bytecode and not the machine native code.

⇒ The bytecode generated is a non-executable code and needs an interpreter to execute on a machine. This interpreter is the JVM and thus the Bytecode is executed by the JVM.

⇒ And finally program runs to give the desired output.

Comparing to C:

In C, the compiler generates an .exe file which is OS dependent. When we try to run this .exe file on another OS it does not run, since it is OS dependent and hence is not compatible with the other OS.

* ~~Exp~~ 5) Explain pass by value and pass-by reference with example.

⇒ pass-by value or call by value means calling a method with a parameter as value. Through this, the argument value is passed to the parameter.

While pass by re-ference or call by reference means

Calling a method with a parameter as a reference. Through this, the argument reference is pass to the parameter.

In pass-by value, the modification done to the parameter passed does not reflect in the caller's scope while pass-by reference, the modification done to the parameter passed are persistent and changes are reflected in the caller's scope.

But java uses only call by value. It creates a copy of references and pass them as value to the methods. If reference contains objects, then the value of an object can be modified in the method but not the entire object.

Example:

```
Public class Example {
    int a = 10;
    void call(int a){
        a += 10;
    }
    psvm( ){
        Example e = new Example();
        e.call(100);
        sout(e.a);
    }
```

Example: pass by reference:

```
public class Example{
  int a=10;
  void call(Example e){
    e.a=10;
  }
psvm(){
  Example e=new Example();
  sout(e.a);
  e.call(e);
  sout(e.a);
}
```

(c)

(1)

Instance variable:

A variable which is declared inside a class and outsi all the methods and block is an instance variable. The general scope of an instance variable is throughout the cla except in static methods. The lifetime of an instance variable is untill the object stays in memory.

class variable:

It is a instance variable with static marked. The general scope of a class variable is throughout the

class and the lifetime of a class or variable is until the end of the program or as long as the class is loaded in memory.

Local variable:

All other variable which are not instance and class variables are treated as local variables including the parameters in a method. Scope of a local variable is within the block in which it is declared and the lifetime of a local variable is until the control leaves the block in which it is declared.

example:

```
Class Sample{
  int x,y;        // x and y are instance variable//
  Static int result; // result is class variable //
  void add(int a, int b) // a and b are local variable//
  {
    x=a; y=b;
  }
  psvm(){
    Sample s= new sample();
    s.add();
  }
```

(iii) when we print an object of a class. output will be -

@classname @ MemoryAdress.

So the answer will be Table@MemoryAddress

(iv)

```
Class Table{

    Public String ~~to~~ toString()
        {
        return " This is a table class ";
        }

Class Main }

    Public static void main( )
        } Table t = new Table();
          sout (t);
    }
```

(3)

(i) Difference between nested class and inner class?

⇒ In Java, we can define a class within another class. Such a class is called Nested class.

Nested class are further divided into two categories.

1. Static nested class.
2. Non static nested class.

The non-static nested classes are called inner classes.

A nested class is a member of its enclosing class and it can be declared public, protected, package or private. Static nested classes do not have access to other members of the enclosing class where as inner classes (or non static inner classes) have access to other members of the enclosing class.

(ii) Overloading occurs when two or more methods in one class have the same method name but different parameters.

Overriding occurs when two methods have the same method name and parameters. One of the methods is in the parent class, and the other is in the child class.

Overloading	Overriding
⇒ Must have at least two method by the same name in the class.	⇒ Must have at least one method by the same name in both parent and child classes.

Overloading

⇒ Must have different number of parameters.

⇒ Overloading is known as compile-time polymorphism.

Overriding

⇒ Must have same number of parameters.

⇒ Overriding is known as runtime polymorphism.

(iii) Thread.sleep() method.

What is synchronization?

⇒ When two or more threads need access to a shared resource, they need some way to ensure that the resource will be used by only one thread at a time. The process by which is this is achieved is called synchronization.

4)

(i) addMouseMotionListener() (for key Listener, addKeyListener() ~~add key listener~~)

(ii) ArrayList

vector

⇒ ArrayList is not synchronized.

⇒ Vector is synchronized.

⇒ ArrayList increases 50% of current array size if the number of elements exceeds from its capacity.

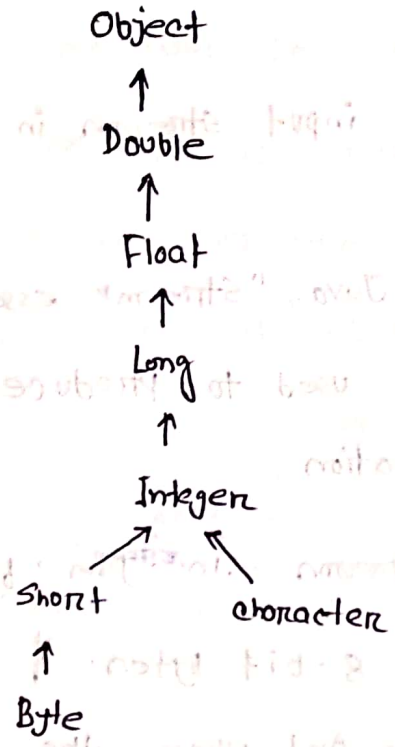
⇒ Vector increments 100% means double the array size if the total number of elements exceeds than its capacity.

⇒ ArrayList is fast because it is non-synchronized.

⇒ Vector is slow because it synchronized.

(v) Show the auto type promotion rules in java.

⇒ The name type promotion specifies that a small size data type can be promoted to a large size data type, i,e, an Integer data type can be promoted to long, float, double, etc. This automatic type promotion is done when any method which accepts a higher size data type argument is called with the smaller data type.



Type promotion hierarchy

✓ Differentiate Set vs Map -

Set	map
⇒ Set is used to construct the mathematical set in java.	⇒ Map is used to do mapping in database
⇒ It can not contain repeated values.	⇒ It can have the same value for different keys.
⇒ we can easily iterate the set elements using the keyset() and entryset() method of it.	⇒ Map element can not be iterated.

(6)

(b) (ii) There are two types of input streams in JAVA. Explain those.

⇒ From the perspective of Java, "Streams" essentially refers to an abstraction that is used to produce and consume flow of sequential information.

There are two types of streams in Java: byte and character. When an I/O stream manage 8-bit bytes of raw binary data, it is called a byte stream. And, when the I/O stream manages 16-bit unicode characters, it is called character stream.

(c)

(i) what is buffer?

⇒ Buffer is a block of memory which we can write data, which we can later be read again. The memory block is wrapped with a NIO buffer object, which provides easier methods to work with the memory block.

Q What is static and why do you use them.

⇒ In java, static keyword is mainly used for memory management. It can be used with variables, methods, blocks and nested classes. It is a keyword which is used to share the same variable or method of a given class. Basically, static is used for a constant variable or a method that is same for every instance of a class.