Marwadi UNIVERSITY

Index

Name. Shashank Bagda
Subject. OOP
Semester 2

Year. 2021/22
Class. 2TK1
Roll No. 92100133020

No.	Date	Experiment Description / Subject	Page	Marks	Faculty's Signature
1	3/3/22	Worksheet - 1			
2	17/3	Worksheet - 3			

Subject Code: 01CT0105

Subject Name: Object Oriented Programming

B. Tech. Year – I (Semester II)

UNIT – 3

Worksheet – 3

Enrollment No: 92100133020

Name: Shashank Bagda

Subject Faculty:

Prof. Kapil Shukla

Information and Communication Technology Department,

Marwadi University

Q-1: What do you mean by Mutable and Immutable?

Answer: Java is object oriented programming language As it is object oriented it's all methods and mechanism revolves around the objects. One object based concept is mutable and immutable in java. Objects in java are either mutable or immutable it depends on how the object can be iterated.

Q-2: Differentiate: String and StringBuffer Class

Answer:

String	StringBuffer
1) String class is immutable	1) String Buffer class is mutable
2) String is slow and consumes more memory when we concatinate too many strings because every time it creates new instance.	2) String Buffer is fast and consumes less memory when we concatenate strings.
3) String class is slower while performing concatenation operation.	3) String Buffer is faster while performing concatenation operation.
4) String class use string constant pool.	4) String Buffer uses heap memory
5) String overrides the equals() method of object class. So you can compare the contents of two strings by equals() method.	5) String Buffer class doesn't override the equals() methods of object class.

Q-3: Explain following String Class methods with its use and syntax.

Answer:

a) charAt():

The java string class charAt() method returns a char value at the given index number. The index number starts from 0 and goes to n-1, where n is the length of the string. It returns StringIndexOutOfBoundException, if the given index number is greater than or equal to this string length or a negative number.

b) compareTo():

The java string class compareTo() method compares the given string with the current string. It returns a positive number negative number or 0. It compares strings on the basis of the unicode value of each characters in the strings.

c) contains():

The java string class contains() methods searches the sequence of characters in this string. It returns true if the sequence of char values is found in this string otherwise returns false

Eg: public boolean contains (CharSequence sequence)

d) endsWith():

The java string class endsWith() method checks if this string ends with a given suffix. It returns true if this string ends with the given suffix, else returns false.

e) indexOf():

The java string class indexOf() methods returns the position of the first occurence of the specified character or string in a specilized string.

f) isEmpty():

The java string class isEmpty() method checks if the input string is empty or not.

g) length():

The java string class length() method finds the length of a string. The length of the Java string is the same as the unicode code units of the string.

h) matches():

This method tells whether or not this string matches the given regular expression. An invocation of this method of the form Str.matches(regex) exactly the same result as the expression Pattern.matches(regex, str).

i) substring():

The java string class substring() method returns a parts of the string we pass beginindex and endindex number position in the Java substring method where beginIndex is inclusive & endIndex is exclusive.

j) trim():

The java string class trim() method diminates leading and trailing spaces. The unicode value of space character is '\u0020'. The trim() method in Java string checks this unicode value before & after the string.

Q-4: Explain Command Line Argument with Example.

Answer:

The command line argument in java allow the programmeros to pass the arguments during the execution of a program. The useros can pass the arguments during the execution by passing the command-line arguments inside the main() method. A command-line arguments is nothing but the information that we pass after typing the name of the Java program during the program execution. These arguments get stored as strings in a string array that is passed to the main() function. We can use these command line arguments as input in our Java

→
```
public class CommandLine
{
   public static void main (String s[])
   {
      for (int i=0 ; i < s.length ; i++)
      {
        System.out.println(" s [" + i + "] : " + s[i] );
      }
   }
}
```

⇒ Output

```
CommandLine : Shashank Bagda
    s[0] : Shashank
    s[1] : Bayda
```

Q-5: What is the use of Wrapper Class? List all primitive data type and its Wrapper class.

Answer:

- They convert primitive data types into objects. Objects are needed if we wish to modify the arguments passed into a method.
- Data structures in the collection framework, such as ArrayList & vector, store only objects and not primitive types.
- The classes in java.util package handles only objects and hence wrappers classes help in this case also.
- An object is needed to support synchronization in multithreading.

Primitive Data Type	Wrappers Class
char	Character
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double
boolean	Boolean

Q-6: What is Boxing and Unboxing in Java? Explain with example.

Answer:

The automatic conversion of primitive data type into the corresponding wrapper class is known as boxing, for example byte to byte, char to characters, int to one Integer, long to Long, float to Float, boolean to Boolean, double to Double and short to Short.

Eg: Primitive to Wrappers

```
public class Wrappers {
{
  public static void main (String [] args)
  {
    int a = 20;
    Integer i = Integer.valueOf (a);
    Integer j = a;
    System.out.println (a + " " + i + " " + j);
  }
}
```

The automatic conversion of wrapper type into its corresponding primitive type is known as unboxing. It is the reverse process of autoboxing. Since Java 5, we do not need to use the intValue() method of wrapper class to convert the wrapper type into primitives.

Q-7: Explain following concept with Example.

a) Class:

A class is a user defined blueprint or prototype from which objects are created. It represents the set of properties or methods that are common to all objects to one type. In general, class declaration can include these components like modifier, class name, class keywords, superclass, Interface, Body.

b) Object:

An entity that has state and behaviour is known as an object eg. chair, bike, markers, pen, table, etc. It can be physical or logical. An object has three characteristic.

1) State = represents the data of an object
2) Behaviour = represent behaviour of Object such as deposit, etc
3) Identity = An object identity is typically implemented via a unique ID. The value of the ID is not visible to external user.

c) Data Member:

Data members may be of any type, including classes already defined, pointers to object of any type, or even references to objects of any type. Data members may be private or public, but are usually held private so that values may only be changed at the discretion of the class function members.

d) Member Function:

Variables declared within a class preceded by a data type which define the state of an object are called data members, and function which define the behaviour of an object of that class are called members function.

e) Constructor:

A constructor is a block of codes similar to the method. It is called when an instance of class is created. At the time of calling constructor, memory for the object is allocated in the memory. Every time an object is created using the new() keyword, at least one constructor is called. It calls a default constructor if there is no constructor available in class.

f) Destructor:

It is a special method that automatically gets called when an object is no longer used when an object completes its life-cycle the garbage collector deletes that object and free the memory occupied by the object.

Q-8: Explain following keywords with Example.

a) static:

The static keyword in java is used for memory management mainly, we can apply static keyword with variables, methods blocks and nested classes. The static keyword belongs to the class that an instance of the class. The static can be : Variable, method, block, nested class.

Eg:
```
class Calculate
{
  static int cube (int x)
  {
    return x*x*x;
  }
  public static void main (string args [])
  {
    int result = Calculate.cube (5);
    System.out.println (result);
  }
}
```

b) this: There can be lot of usage of Java this keyword. In this, is a reference variable that refers to the current object.

Eg:
```
Class A
{
  void m() { System.out.println ("Hello m"); }
  void n() { System.out.println ("Hello n");
             this.m()
           }
Class TestThis
{
  public static void main (string args[])
  {
    A a = new A();
    a.n();
  }
}
```

```
Int i = a.intValue();
Int j = a
System.out.println (a+" "+i+" "+j)
}
}
```

c) final: [class | method | variable]

Final Class:

When a class is declared with final keyword, it is called a final class. A final class cannot be extended (inherited).

Uses: ① One is definitely to prevent inheritance, as final classes cannot be extended. ② To create an immutable class like the predefined String class.

Eg:
```
final class A
{
    // methods
}
```

Final Method

When method is declared with final keyword, it is called a final method. A final method cannot be overridden. The object class does this - a number of its methods are final. We must declare methods with the final keyword for which we required to follow this...

Eg:
```
Class A
{
    final void m1()
    {
        System.out.println("  ");
    }
}
```

Final Variable

When a variable is declared with final keyword, its value can't be modified, essentially, a constant. This also means that you must initialize a final variable. If the final variable is a reference, this means that the variable cannot be re-bound to reference another object, but the internal state of the object pointed by that reference variable can be changed.

Eg: `final int THRESHOLD = 5;`

d) abstract: [class | method]

A class which is declared with the abstract keyword is known as abstract class in Java. It can have abstract and non-abstract methods.

A class which is declared as abstract is known as abstract class. It can have abstract and non-abstract methods. It needs to be extended and its method implemented. It cannot be instantiated.

- An abstract class must be declared with an abstract keyword. It can have abstract and non-abstract methods. It cannot be instantiated. - It can have constructors and static methods also.

Abstract method can only be used in an abstract class, and it does not have a body. The body is provided by the sub class (inherited form).

Eg:

```
abstract class Animal
{
    public abstract void animalSound();
    public void sleep()
    {
        System.out.println("Zzz");
    }
}
```

Q-9: Differentiate:

Overloading	Overriding
1) When two or more methods in the same class have the same name but different parameters, it's called overloading.	1) When the method signature name and parameters are the same in the superclass and the child class it's called overriding.
2) Overloading implements compile time polymorphism.	2) Overriding implements runtime polymorphism.
3) Overloading occurs between the methods in the same class.	3) Overriding methods have the same signature i.e. same name and method arguments.
4) Overloading the method to call is determined at the compile-time.	4) Overriding the method call as determined at the runtime based on the object type.
5) Overloading breaks, the compile time errors will come and it's easy to fix.	5) Overriding breaks, it can cause serious issues in our program because the effect will be visible at runtime.
6) Overloading method names are the same but the parameters are different.	6) The method overriding occurs between superclass and subclass.

Q-10: Explain following visibility modifiers.

a) private:

The private modifier specifies that the member can only be accessed in its own class. The protected modifier specifies that the member can only be accessed within its own package and, in addition by a subclass of its class in another package.

b) default:

When we don't use any password explicitly java will set a default access to a given class, method or property. The default access modifier is also called package-private, which means that all members are visible within the same package but aren't accessible from others packages.

c) protected:

The protected access modifier is accessible within package and outside the package but through inheritance only. The protected access modifiers can be applied on the data members, method and constructors. It can't be applied on the class.

d) public:

The public access modifier is specified using the keyword public. The public access modifier has the widest scope among all other access modifiers. Classes, methods or data members that are declared as public are accessible from everywhere in the program. There is no restriction on the scope of public data members.

Q-11: Explain Garbage Collection process using example.

Java garbage collection is the process by which java program perform automatic memory management. Java programs compile to bytecode that can be run on a Java Virtual Machine, or JVM for short. When Java programs run on the JVM, objects are created on the heap, which is a portion of memory dedicated to the program. Eventually, some objects will no longer be needed. The garbage collector finds these unused objects and ~~deletes~~ them to free up memory. There are different method to unreferenced an object :

① By nulling the reference

→ Employee e = new Employee ();
e = null;

② By assigning a reference to another

→ Employee e1 = new Employee ();
Employee e2 = new Employee ();
e1 = e2;

③ By anonymous object;

→ new Employee ();

References

Q No	Book Name	Page No
1	Programming in Java	162
2	Programming in Java	164
3	Online & Thinking in Java	511
4	Java 8	21
5	Programming in Java / Javas	66
6	Online	-
7	Online	-
8	Geeks for Geeks / Javat.com	-
9	Online	-
10	Geeks for Geeks / Java 8	-
11	Thinking in Java / Javat.com	-