21/03937 MANYONI MAGDALINE OKENYURI

DIPLOMA IN INFORMATION TECHNOLOGY

DIT 409– JAVA PROGRAMMING

TASK 3:

1. Explain the differences between primitive and reference data types.

When a primitive variable is declared, the computer allocates some memory to store the value that it has assigned. The size of the container that's reserved depends on the type of primitive.

It can be used in data management to define the characteristics of an identifier. Whether the data is externally mandated or internally authored, it's unambiguous and non-negotiable.

2. Define the scope of a variable (hint: local and global variable)

A scope is a region of the program and broadly speaking there are three places, where variables can be declared: Inside a function or a block which is called local variables, In the definition of function parameters which is called formal parameters. Outside of all functions which is called global variables

3. Why is initialization of variables required?

In order to give a correct initial value to a variable, Java must first do something about it. This is very important to do, as an error might occur, or it might just be a case that you haven't updated the variable. Most of the time, Java will tell you to start the variable, even if an error has already been caused.

4. Differentiate between static, instance and local variables.

Static variables are created when the program starts and destroyed when the program stops. Instance variables are created when an object is created with the use of the keyword 'new' and destroyed when the object is destroyed. Local variable is defined as a type of variable declared within programming blocks or subroutines.

5. Differentiate between widening and narrowing casting in java.

Widening conversions preserve the source value but can change its representation. This occurs if you convert from an integral type to Decimal, or from Char to String. A narrowing conversion changes a value to a data type that might not be able to hold some of the possible values.

6. the following table shows data type, its size, default value and the range. Filling in the missing values.

TYPE	**SIZE (IN BYTES)**	**DEFAULT**	**RANGE**
boolean	1 bit	False	true, false
Char	2	'**\u0000**'	'\0000' to '\ffff'
Byte	2^8	0	-2^7 to $+2^7-1$
Short	2 bytes	0	-2^{15} to $+2^{15}-1$
Int	4	0	-2^{31} to $+2^{31}-1$
Long	8	0L	-2^{63}
Float	4	00.0f	*3.4E-38 to 3.4E+38*
Double	8	**0.0d**	-1.8E+308 to +1.8E+308

7. Explain the importance of using Java packages

Packages help prevent naming conflicts. It orders the classes according to their functions. Hence it is easy to search for classes. You can control the accessibility of the classes by using access specifiers and packages together

8. Explain three controls used when creating GUI applications in Java language.

A GUI program consists of three types of software: Graphical Components that make up the Graphical User Interface. Listener methods that receive the events and respond to them. Application methods that do useful work for the user.

9. Explain the difference between containers and components as used in Java.

Containers are the interface between a component and the low-level, platform-specific functionality that supports the component. Before it can be executed, a web, enterprise bean, or application client component must be assembled into a Java EE module and deployed into its container.

A Component is a class or function that describes part of a React UI. Container is an informal term for a React component that is connect -ed to a redux store.

10. Write a Java program to reverse an array having five items of type int.

```
public class reverseArray {

  static void reverse (int a [], int n)

  {

  int[] b = new int[n];

  int j = n;

  for (int i = 0; i < n; i++) {

  b[j - 1] = a[i];

                    j = j - 1;

            }

            // printing the reversed array

System.out.println("Reversed array is: \n");

            for (int k = 0; k < n; k++) {

                    System.out.println(b[k]);

            }

  }
```

```
    public static void main(String[] args)
    {
        int [] arr = {10, 20, 30, 40, 50};
        reverse(arr, arr.length);
    }
}
```

11. Programs written for a graphical user interface have to deal with “events.”
    i) Explain what is meant by the term event.

An event can be defined as changing the state of an object or behavior by performing actions. Actions can be a button click, cursor movement, keypress through keyboard or page scrolling, etc. The java. awt. event package can be used to provide various event classes.

    ii) Give at least two different examples of events, and discuss how a program might respond to those events.

12. Explain the difference between the following terms as used in Java programming.
    i) Polymorphism and encapsulation

Polymorphism ensures that the proper method will be executed based on the calling object's type. Encapsulation allows you to control access to your object's state, while making it easier to maintain or change your implementation at a later date.

    ii) Method overloading and method overriding

Overriding occurs when the method signature is the same in the superclass and the child class while Overloading occurs when two or more methods in the same class have the same name but different parameters.

    iii) Class and interface

Difference between an interface and an abstract class is that an interface cannot have state, whereas the abstract class can have state with instance variables.

    iv) Inheritance and polymorphism

Polymorphism means "many forms", and it occurs when we have many classes that are related to each other by inheritance. Inheritance lets us inherit attributes and methods from another class. Polymorphism uses those methods to perform different tasks.

13. Using examples, explain the two possible ways of implementing polymorphism. Show your code in java.

We can achieve polymorphism in Java using the following ways:

1.Method overriding

Overriding occurs when the method signature is the same in the superclass and the child class

```
class Language {
  public void displayInfo() {
    System.out.println("Common English Language");
  }
}

class Java extends Language {
  @Override
  public void displayInfo() {
    System.out.println("Java Programming Language");
  }
}

class Main {
  public static void main(String[] args) {
```

```
    // create an object of Java class
    Java j1 = new Java();
    j1.displayInfo();

    // create an object of Language class
    Language l1 = new Language();
    l1.displayInfo();
  }
}
```

2.Method Overloading

Overloading occurs when two or more methods in the same class have the same name but different parameters.

```
class Pattern {

  // method without parameter
  public void display() {
    for (int i = 0; i < 10; i++) {
      System.out.print("*");
    }
  }

  // method with single parameter
  public void display(char symbol) {
    for (int i = 0; i < 10; i++) {
      System.out.print(symbol);
    }
```

```
  }
}

class Main {
  public static void main(String[] args) {
    Pattern d1 = new Pattern();

    // call method without any argument
    d1.display();
    System.out.println("\n");

    // call method with a single argument
    d1.display('#');
  }
}
```

Mutable classes.

Explain what is meant by mutable class

A mutable value is one that can be changed without creating an entirely new value. In JavaScript, objects and arrays are mutable by default, but primitive values are not — once a primitive value is created, it cannot be changed, although the variable that holds it may be reassigned.

Write a program that implements the concept of mutable class

```
1. public class JtpExample {
2.     private String s;
3.     JtpExample(String s) {
```

```
    this.s = s;
    }
    public String getName() {
    return s;
    }
    public void setName(String coursename) {
        this.s = coursename;
        }
        public static void main(String[] args) {
        JtpExample obj = new JtpExample("JavaTpoint");
        System.out.println(obj.getName());
      // Here, we can update the name using the setName method.
      obj.setName("Java Training");
        System.out.println(obj.getName());
        }
        }
```

b. Immutable classes.

Explain what is meant by immutable class

Immutable class in java means that **once an object is created, we cannot change its content**. In Java, all the wrapper classes (like Integer, Boolean, Byte, Short) and String class is immutable. We can create our own immutable class as well.

Write a program that implements the concept of immutable class

```
public class JtpExample1 {
    private final String s;
    JtpExample1(final String s) {
    this.s = s;
    }
    public final String getName() {
    return s;
    }
    public static void main(String[] args) {
```

```
10.         JtpExample obj = new JtpExample("Core Java Training");
11.         System.out.println(obj.getName());
12.         }
13.         }
```

c. Explain the situations where mutable classes are more preferable than immutable classes when writing a Java program.

2. Explain what a String buffer class is as used in Java

A string buffer is like a String, but can be modified. At any point in time it contains some particular sequence of characters, but the length and content of the sequence can be changed through certain method calls. String buffers are safe for use by multiple threads

The syntax of creating an object of StringBuffer class

Java StringBuffer class is used to create mutable (modifiable) String objects.
...
Important Constructors of StringBuffer Class.

Explain the methods in the StringBuffer class

public synchronized StringBuffer	append(String s)
public synchronized StringBuffer	insert(int offset, String s)
public synchronized StringBuffer	replace(int startIndex, int endIndex, String str)
public synchronized StringBuffer	delete(int startIndex, int endIndex)