21/02602
Shallet Mwende

Task3

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

- Primitive data types specify the size and type of variable values while reference data types specify the reference/address of the variable values.
- Primitive data types store values while reference data types store objects in heap space.
- Primitive data types are already defined in Java while reference data types can be defined by the user.
- Primitive data types are basic types of data eg,char,int.. while reference data are any instantiable class as well as arrays eg, string,Scanner..

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

Outside of all functions are called global variables and inside a function is called local variables.

3. Why is initialization of variables required?

- To avoid a value having an unknown value.
- So that they can be used in a program.

4. Differentiate between static, instance and local variables.

Local variables are defined within a method or a code block,Instance variables are defined outside a method at the class level and Static variables are defined outside a method at the class level.

5. Differentiate between widening and narrowing casting in java.

Widening conversion changes a value to a data type that can allow for any possible value of the original data while 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	1	0	-27 to +27-1
Short	2	0	-215 to +215-1
Int	4	0	-231 to +231-1
Long	8	0L	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
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

- Used to group related classes.
- Used to write a better maintainable code.
- Used to avoid name conflicts.

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

Label - Is used to provide a descriptive text string that cannot be changed directly by the user.

Checkbox - Used to display options to the user, where the user can select more than one option.

Button - Used to execute blocks of code in a program when clicked by the user.

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

Containers can have other containers and components in it while components cannot have other components in them.

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

```
import java.util.*;
import java.util.stream.*;
public class PrintArrays
{
  public static void main(String[] args) {
  //creating the array with 5 items
  Integer[] myArray = { 1, 2, 3, 4, 5};

  //print the array starting from last element
    for(int i=myArray.length-1;i>=0;i--) {
      System.out.print(myArray[i] + "  ");
    }
  }
}
```

11. Programs written for a graphical user interface have to deal with “events.” Explain what is meant by the term event.
Give at least two different examples of events, and discuss how a program might respond to those events.

Event – This is an object that is created when something changes in a graphical user interface.
Example: Action event- Represents a graphical event is clicked like a button or item a list.

12. Explain the difference between the following terms as used in Java programming.

Polymorphism and encapsulation

Polymorphism ensures that the proper method will be executed based on the calling object's type while 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.

Method overloading and method overriding

Method overloading is when two or more methods in the same class have the same name but different parameters while method overriding is when the method name and parameters are in the same super class and child class.

Class and interface

A class can inherit another class but an interface cannot inherit a class.

Inheritance and polymorphism

Inheritance supports the concept of reusability and reduces code length in object-oriented programming while polymorphism allows the object to decide which form of the function to implement at compile-time as well as run-time.