java

Assignment answers

1.

Primitive data types are basic data types in programming languages that represent simple values, such as numbers, characters, and Booleans/binary. Reference data types, on the other hand, are more complex data types that store memory addresses that refer to objects or arrays.

1. Scope- part of the program where the variable can be accessed

Local variable is declared within a block of code and can only be accessed within that block.

Global variable is declared outside of any function and can be accessed anywhere within the program.

1. Initialization of variables is required to give them an initial value before they are used in the program which prevents the program from accessing garbage values, which can lead to unexpected behavior or errors.
2. Static variables are declared using the keyword "static" and maintain their value across all instances of a class.

Instance variables are declared within a class and are unique to each instance of the class.

Local variables are declared within a method or block of code and have a limited scope.

1. Widening casting – conversion of smaller data type to a larger data type like int to long which can be done completely and without the risk of loss of data

Narrowing casting - conversion of a larger data type to a smaller data type like as long to int which must be done explicitly and may result in a loss of data

1.

TYPE	SIZE (IN BYTES)	DEFAULT	RANGE
Boolean	1 bit	false	true, false
char	2	'\u0000'	'\u0000' to '\uffff'
byte	1	0	-128 to 127
short	2	0	-32,768 to 32,767
int	4	0	-2,147,483,648 to 2,147,483,647
long	8	0L	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
float	4	0.0f	1.4E-45 to 3.4028235E38
double	8	0.0d	4.9E-324 to 1.7976931348623157E308

1. A package - way of organizing related classes and interfaces into a single namespace to avoid naming conflicts and make the code more modular and reusable.
2. Java packages offer several important benefits in programming, including:

- Encapsulation: Packages provide a way to hide implementation details and only expose a specific set of interfaces to other parts of the code, improving the overall organization and maintainability of the code.
- Namespace management: By grouping related classes and interfaces into a package, Java ensures that no two classes with the same name can exist in the same package, preventing naming conflicts and increasing clarity and consistency in the code.
- Code reuse: By allowing developers to group related code into functional packages, Java simplifies code reuse, making it easier to import and use common functionality across projects and teams.
- Accessibility control: Packages can be designated as public or private, allowing developers to control which interfaces and classes can be accessed from other parts of the code, reducing the risk of security vulnerabilities or accidental misuse of code.

section 2

1.

```
import java.util.Scanner;

public class SurnameAndAge {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        System.out.print("Enter your surname: ");
        String surname = input.nextLine();

        System.out.print("Enter your age: ");
        int age = input.nextInt();

        System.out.println("The number of characters in your surname is " + surname.length());

        if (age % 2 == 0) {
            System.out.println("Your current age is an even number");
        } else {
            System.out.println("Your current age is an odd number");
        }
    }
}
```

2.

```
import java.util.Scanner;

public class AverageMarks {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        System.out.print("Enter marks for unit 1: ");
        double unit1 = input.nextDouble();

        System.out.print("Enter marks for unit 2: ");
        double unit2 = input.nextDouble();

        System.out.print("Enter marks for unit 3: ");
        double unit3 = input.nextDouble();
```

```
        System.out.print("Enter marks for unit 4: ");
        double unit4 = input.nextDouble();

        System.out.print("Enter marks for unit 5: ");
        double unit5 = input.nextDouble();

        double average = (unit1 + unit2 + unit3 + unit4 + unit5) / 5;

        System.out.printf("The average marks is %.2f", average);
    }
}
```

3.

```
import java.util.Scanner;

public class DivisibilityTest {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        System.out.print("Enter a number: ");
        int number = input.nextInt();

        for (int i = 0; i <= 9; i++) {
            if (number % i == 0) {
                System.out.println("The number is divisible by " + i + " because it ends with a " + number % 10);
            }
        }
    }
}
```

4.

```
public class Multiples {
    public static void main(String[] args) {
        for (int i = 71; i <= 150; i++) {
            if (i % 2 == 0) {
                System.out.println(i + " is a multiple of 2");
            }
            if (i % 3 == 0) {
                System.out.println(i + " is a multiple of 3");
            }
            if (i % 7 == 0) {
                System.out.println(i + " is a multiple of 7");
            }
        }
    }
}
```

5.

```
import java.util.Scanner;

public class Calculator {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
```

```
        System.out.print("Enter a number: ");
        double num1 = input.nextDouble();

        System.out.print("Enter an operation (+, -, *, /): ");
        String operation = input.next();

        System.out.print("Enter another number: ");
        double num2 = input.nextDouble();

        double result;

        switch (operation) {
            case "+":
                result = num1 + num2;
                break;
            case "-":
                result = num1 - num2;
                break;
            case "*":
                result = num1 * num2;
                break;
            case "/":
```