AHMED KHADAR AHMED SCT221-0780/2022

Java programming

Exercise 1 of 16

Instructions:
All programs should be written, and linked to an online repository like GitHub.
A video to get you started with GitHub has been posted on Moodle.
After completing your assignment, post the link on the link on Moodle. An instructor will follow the posted link to access and grade your work.
Note that: Your program should always be well-commented. At the top of your source code file, you should write a short description of what your program does and add other comments to help in explaining your code.
All of your variables should be given a deceptive name. Avoid giving your variables names like a, b, I, x, y etc.
In case you copy your friend's work, you both get a Zero (0).

Section 1:

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

ANSW:

Primitive data types: These are the basic data types in Java, such as int, double, boolean, etc. They store values directly and are stored in the stack memory.

Reference data types: These are objects that represent complex data structures, such as arrays, strings, and user-defined classes. They store the memory address of the object in the stack and the actual object data in the heap memory.

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

ANSW:

Scope of a variable:

Local variable: A variable declared within a method or a block of code. Its scope is limited to that method or block.

Global variable (or class variable): A variable declared at the class level, outside any method. Its scope is the entire class.

3. Why is initialization of variables required.

   ANSW:

   (i)Prevent Undefined Behavior: Uninitialized variables can contain garbage values leading to unpredictable behavior.

(ii)Ensure Predictability: Initializes variables to known values, ensuring that operations on them produce expected results.

(iii)Avoid Runtime Errors: Uninitialized variables can cause runtime errors, especially in languages where default values are not automatically assigned.

4. Differentiate between static, instance and local variables.

ANSW:

Static variables: Belong to the class and are shared among all instances of the class. They are initialized when the class is loaded.

Instance variables: Belong to individual objects of a class. Each object has its own copy of the instance variables.

Local variables: Declared within a method or a block of code. They are accessible only within the scope in which they are defined.

5. Differentiate between widening and narrowing casting in java.

A Widening (implicit) casting: Conversion from a smaller data type to a larger data type, e.g., int to double. This is done automatically by the compiler and is safe.

Narrowing (explicit) casting: Conversion from a larger data type to a smaller data type, e.g., double to int. This can result in a loss of precision and must be done explicitly by the programmer.

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	'\0000'	‘\0000’ to ‘\ffff’
Byte	1	0	-128 to +127
Short	2	0	-2^{15} to $+2^{15}-1$
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	00.0f	-3.4028235E+38 to +3.4028235E+38
Double	8	0.0	-1.8E+308 to +1.8E+308

7. Define class as used in OOP

ANSW:

is a blueprint or a template for creating objects. It defines the properties (data) and behaviors (methods) that the objects of the class will have.

8. Explain the importance of classes in Java programming.

ANSW:
(i)Modularity: Classes break down complex systems into manageable parts.
(ii)Reusability: Classes can be reused across different projects.
(iii)Encapsulation: Classes hide implementation details and expose only necessary methods.
(iv)Inheritance: Classes enable inheritance, allowing one class to inherit properties from another.

Section 2:

1. Write a Java program that asks the user to enter their sur name and current age then print the number of characters of their sir name and even or odd depending on their age number.

Example of Expected result:
If sir name is Saruni and age is 29, output will be;
then the number of characters is 6.
Your current age is an odd number

ANSW:

```
import java.util.Scanner;

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

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

        System.print("Enter your current age: ");
        int age = scanner.nex import java.util.Scanner;

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

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

        System.print("Enter your current age: ");
        int age = scanner.nextInt();

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

        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.");
        }
    }
```

```
}tInt();

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

        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. Write Java program to ask student to enter the marks of the five units they did last semester, compute the average and display it on the screen. (Average should be given in two decimal places).

ANSW:

```
import java.util.Scanner;


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


        System.out.print("Enter the marks for Unit 1: ");
        double unit1 = scanner.nextDouble();


        System.out.print("Enter the marks for Unit 2: ");
        double unit2 = scanner.nextDouble();


        System.out.print("Enter the marks for Unit 3: ");
        double unit3 = scanner.nextDouble();


        System.out.print("Enter the marks for Unit 4: ");
        double unit4 = scanner.nextDouble();


        System.out.print("Enter the marks for Unit 5: ");
        double unit5 = scanner.nextDouble();
```

```
        double sum = unit1 + unit2 + unit3 + unit4 + unit5;
        double average = sum / 5;

        System.out.printf("The average of the five units is: %.2f\n", average);
    }
}
```

3. Write a program that will help kids learn divisibly test of numbers of integers. The program should check whether the given integer is divisible by integers in the range of 0-9. For example, if a number (955) is divisible by five, the program should print, the number is divisible by 5 because it ends with a 5, and 900 is divisible by 5 because it ends with a 0(zero).

ANSW:

```
import java.util.Scanner;

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

        // Prompt the user to enter an integer
        System.out.print("Enter an integer: ");
        int number = scanner.nextInt();

        // Check divisibility for integers in the range 1-9
        for (int i = 1; i <= 9; i++) {
            if (isDivisibleBy(number, i)) {
                System.out.println("The number " + number + " is divisible by " + i + ".");
                if (i == 2) {
                    if (number % 2 == 0) {
                        System.out.println("The number " + number + " is divisible by 2 because it is
even.");
                    }
```

```
        } else if (i == 3) {
            if (sumOfDigits(number) % 3 == 0) {
                System.out.println("The number " + number + " is divisible by 3 because the sum of its digits is divisible by 3.");
            }
        } else if (i == 4) {
            if (number % 100 % 4 == 0) {
                System.out.println("The number " + number + " is divisible by 4 because the last two digits form a number that is divisible by 4.");
            }
        } else if (i == 5) {
            if
```

4. Write a Java program to display all the multiples of 2, 3 and 7 within the range 71 to 150.

ANSW:

```
public class MultiplesDisplay {
    public static void main(String[] args) {
        System.out.println("Multiples of 2 within the range 71 to 150:");
        for (int i = 71; i <= 150; i++) {
            if (i % 2 == 0) {
                System.out.println(i);
            }
        }

        System.out.println("\nMultiples of 3 within the range 71 to 150:");
        for (int i = 71; i <= 150; i++) {
            if (i % 3 == 0) {
                System.out.println(i);
            }
        }

        System.out.println("\nMultiples of 7 within the range 71 to 150:");
        for (int i = 71; i <= 150; i++) {
```

```
            if (i % 7 == 0) {
                System.out.println(i);
            }
        }
    }
}
```

5. Create a calculator using java to help user perform the basic operations (+, -, * and /).
    a. User should be asked to enter a number, then an operation, the program computes the operation and display the output to the computer screen.

ANSW:

```
import java.util.Scanner;


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


        // Prompt the user to enter an integer
        System.out.print("Enter an integer: ");
        int number = scanner.nextInt();


        // Check divisibility for integers in the range 1-9
        for (int i = 1; i <= 9; i++) {
            if (isDivisibleBy(number, i)) {
                System.out.println("The number " + number + " is divisible by " + i + ".");
                if (i == 2) {
                    if (number % 2 == 0) {
                        System.out.println("The number " + number + " is divisible by 2 because it is
even.");
                    }
                } else if (i == 3) {
                    if (sumOfDigits(number) % 3 == 0) {
```

```
                System.out.println("The number " + number + " is divisible by 3 because the sum of its digits is divisible by 3.");
            }
        } else if (i == 4) {
            if (number % 100 % 4 == 0) {
                System.out.println("The number " + number + " is divisible by 4 because the last two digits form a number that is divisible by 4.");
            }
        } else if (i == 5) {
            if (number % 10 == 0 || number % 10 == 5) {
                System.out.println("The number " + number + " is divisible by 5 because it ends with a " + (number % 10) + ".");
            }
        } else if (i == 6) {
            if (number % 2 == 0 && sumOfDigits(number) % 3 == 0) {
                System.out.println("The number " + number + " is divisible by 6 because it is divisible by both 2 and 3.");
            }
        } else if (i == 7) {
            // For 7, no simple rule is included
        } else if (i == 8) {
            if (number % 1000 % 8 == 0) {
                System.out.println("The number " + number + " is divisible by 8 because the last three digits form a number that is divisible by 8.");
            }
        } else if (i == 9) {
            if (sumOfDigits(number) % 9 == 0) {
                System.out.println("The number " + number + " is divisible by 9 because the sum of its digits is divisible by 9.");
            }
        }
    }
}
```

```
    private static boolean isDivisibleBy(int number, int divisor) {
        return number % divisor == 0;
    }

    private static int sumOfDigits(int number) {
        int sum = 0;
        while (number != 0) {
            sum += number % 10;
            number /= 10;
        }
        return sum;
    }
}
```

ATM CODE

1: create a java program to store an ATM. the program should have 4 main functionalities. (i) check balance (ii) withdraw money (iii) transfer money to another account , cannot be within the same bank where transfer charges is 0 or to another bank where transfer are charges are 0.001 of the amount being transfered (vi) deposit/ received money . you should not send / withdraw more than what you have

ANSW:

```
import java.util.HashMap;
import java.util.Scanner;

public class ATM {
    private static final double TRANSFER_CHARGE_WITHIN_BANK = 0.0;
    private static final double TRANSFER_CHARGE_BETWEEN_BANKS = 0.001;

    private static final HashMap<String, Double> accounts = new HashMap<>();

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

        while (true) {
            System.out.println("Welcome to the ATM!");
```

```
        System.out.println("Please select an option:");
        System.out.println("1. Check Balance");
        System.out.println("2. Withdraw Money");
        System.out.println("3. Transfer Money");
        System.out.println("4. Deposit Money");
        System.out.println("5. Exit");

        int choice = scanner.nextInt();
        scanner.nextLine(); // consume newline character

        switch (choice) {
            case 1:
                checkBalance();
                break;
            case 2:
                withdrawMoney(scanner);
                break;
            case 3:
                transferMoney(scanner);
                break;
            case 4:
                depositMoney(scanner);
                break;
            case 5:
                System.out.println("Goodbye!");
                return;
            default:
                System.out.println("Invalid choice. Please try again.");
                break;
        }
    }
}

private static void checkBalance() {
    System.out.println("Your current balance is: $" + getBalance());
}

private static void withdrawMoney(Scanner scanner) {
    System.out.print("Enter the amount you want to withdraw: $");
    double amount = scanner.nextDouble();
    scanner.nextLine(); // consume newline character

    if (amount > getBalance()) {
        System.out.println("Insufficient funds.");
        return;
```

```
        }

        updateBalance(-amount);
        System.out.println("Withdrawal successful. Your new balance is: $" +
getBalance());
    }

    private static void transferMoney(Scanner scanner) {
        System.out.print("Enter the account number to transfer to: ");
        String toAccount = scanner.nextLine();
        System.out.print("Enter the amount to transfer: $");
        double amount = scanner.nextDouble();
        scanner.nextLine(); // consume newline character

        if (amount > getBalance()) {
            System.out.println("Insufficient funds.");
            return;
        }

        double charge = toAccount.startsWith(getAccountNumber()) ?
TRANSFER_CHARGE_WITHIN_BANK :
TRANSFER_CHARGE_BETWEEN_BANKS;
        double totalAmount = amount + (amount * charge);

        if (totalAmount > getBalance()) {
            System.out.println("Insufficient funds to cover the transfer charge.");
            return;
        }

        updateBalance(-totalAmount);
        updateAccountBalance(toAccount, amount);
        System.out.println("Transfer successful. Your new balance is: $" +
getBalance());
    }

    private static void depositMoney(Scanner scanner) {
        System.out.print("Enter the amount you want to deposit: $");
        double amount = scanner.nextDouble();
        scanner.nextLine(); // consume newline character

        updateBalance(amount);
        System.out.println("Deposit successful. Your new balance is: $" +
getBalance());
    }
```

```
    private static double getBalance() {
        return accounts.getOrDefault(getAccountNumber(), 0.0);
    }

    private static void updateBalance(double amount) {
        double currentBalance = getBalance();
        accounts.put(getAccountNumber(), currentBalance + amount);
    }

    private static void updateAccountBalance(String account, double amount) {
        double currentBalance = accounts.getOrDefault(account, 0.0);
        accounts.put(account, currentBalance + amount);
    }

    private static String getAccountNumber() {
        // Assuming the account number is hardcoded for simplicity
        return "1234567890";
    }
}
```