**Java programming**

**Exercise 1 of 16**

**ANN BAGENI**

**23/03634**

**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.

**Primitive Data Types**:

* These are the most basic data types in Java and store simple values.
* Examples include int, char, Boolean, float, double, etc.
* Primitive types hold their value directly in memory.
* They are not objects, so they don't have methods.

**Reference Data Types**:

* These refer to objects and arrays in Java.
* Reference variables store addresses (references) of objects.
* Examples include String, Arrays, Classes, Interfaces, etc.
* They can have methods that operate on data stored in the object.

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

**Local Variables**:

* Declared inside a method, constructor, or block.
* Can only be accessed within that method, constructor, or block.
* They do not exist after the method exits.

**Global (Instance/Class) Variables**:

* Instance variables are declared inside a class but outside of any method.
* They belong to the object and exist as long as the object is alive.
* Class variables are declared with the static keyword and belong to the class, not any specific object.

1. Why is initialization of variables required?

* Variables must be initialized to give them a defined value before they are used. Uninitialized variables can lead to errors or undefined behaviour in a program.

1. Differentiate between static, instance and local variables.

**Static Variables**:

* Declared with the static keyword.
* Belong to the class rather than any specific instance.
* Shared across all instances of the class.

**Instance Variables**:

* Belong to individual instances (objects) of a class.
* Each object has its own copy of the instance variables.

**Local Variables**:

* Declared inside a method or block.
* Only accessible within the method or block where they are declared.

1. Differentiate between widening and narrowing casting in java.

**Widening Casting**:

* Automatically done by Java.
* Converts a smaller data type to a larger data type (e.g., int to long or float to double).

**Narrowing Casting**:

* Done manually.
* Converts a larger data type to a smaller one (e.g., double to float or long to int).

1. 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 | -128 to 127 |
| Short | 2 | 0 | -215 to +215-1 |
| Int | 4 | 0 | -2^31 to 2^31 - 1 |
| Long | 8 | 0L | - -2^63 to 2^63 - 1 |
| Float | 4 | 00.0f | 1.4E-45 to 3.4028235E+38 |
| Double | 8 | 0.0d | -1.8E+308 to +1.8E+308 |

1. Define class as used in OOP

A class is a blueprint or template from which objects are created. It encapsulates data for the object and defines methods to operate on that data.

1. Explain the importance of classes in Java programming.

Classes provide structure to code and promote reusability.

They allow encapsulation, inheritance, and polymorphism, which are key concepts of object-oriented programming.

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

import java.util.Scanner;

public class SurnameAgeChecker {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Ask user for surname and age

System.out.print("Enter your surname: ");

String surname = scanner.nextLine();

System.out.print("Enter your age: ");

int age = scanner.nextInt();

// Calculate the number of characters in surname

int surnameLength = surname.length();

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

// Check if age is even or odd

if (age % 2 == 0) {

System.out.println("Your age is an even number.");

} else {

System.out.println("Your age is an odd number.");

}

scanner.close();

}

}

1. 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).

import java.util.Scanner;

public class AverageMarks {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

double total = 0;

int numberOfUnits = 5;

// Ask user for marks of five units

for (int i = 1; i <= numberOfUnits; i++) {

System.out.print("Enter the marks for unit " + i + ": ");

total += scanner.nextDouble();

}

// Calculate the average

double average = total / numberOfUnits;

System.out.printf("The average marks are: %.2f%n", average);

scanner.close();

}

}

1. 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).

import java.util.Scanner;

public class DivisibilityChecker {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Ask user for a number

System.out.print("Enter an integer: ");

int number = scanner.nextInt();

// Check divisibility for numbers 1 to 9

for (int i = 1; i <= 9; i++) {

if (number % i == 0) {

System.out.println(number + " is divisible by " + i);

}

}

scanner.close();

}

}

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

public class MultiplesFinder {

public static void main(String[] args) {

System.out.println("Multiples of 2, 3, and 7 between 71 and 150:");

for (int i = 71; i <= 150; i++) {

if (i % 2 == 0 || i % 3 == 0 || i % 7 == 0) {

System.out.println(i);

}

}

}

}

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

import java.util.Scanner;

public class SimpleCalculator {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Ask user for two numbers

System.out.print("Enter the first number: ");

double num1 = scanner.nextDouble();

System.out.print("Enter an operation (+, -, \*, /): ");

char operation = scanner.next().charAt(0);

System.out.print("Enter the second number: ");

double num2 = scanner.nextDouble();

double result;

// Perform the operation

switch (operation) {

case '+':

result = num1 + num2;

break;

case '-':

result = num1 - num2;

break;

case '\*':

result = num1 \* num2;

break;

case '/':

if (num2 != 0) {

result = num1 / num2;

} else {

System.out.println("Error! Division by zero.");

return;

}

break;

default:

System.out.println("Invalid operation.");

return;

}

// Display the result

System.out.println("The result is: " + result);

scanner.close();

}

}