**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

* **Primitive data** types in Java are the basic types of data built into the language. They include byte, short, int, long, float, double, boolean, and char. They hold their values directly and are stored in stack memory.
* **Reference data types**, on the other hand, are objects that are created from class declarations. They hold references (or addresses) to the objects they refer to, rather than holding the object’s value directly. Reference data types are stored in heap memory.

**Differences:**

* **Storage**: Primitives are stored directly in memory, while references store an address to the memory location where the actual object resides.
* **Default Values**: Primitives have default values.
* **Size**: Primitives have a defined size.
* **Operations**: Operations on primitives are generally faster as they deal with direct values.

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

* **Local Variables**: Declared within a method or block and can only be accessed within that method or block.
* **Global Variables**: Also known as class or member variables, they are declared outside any method and can be accessed by any method in the class, and outside the class if they are public.

1. Why is initialization of variables required.

**In Definite Value**: It ensures that the variable has a definite value before it is used, preventing unpredictable behaviours or errors during runtime.

**During Program Correctness**: It helps in maintaining the correctness of the program by providing known starting values for variables.

**Instance of Memory Management**: In some cases, it can also help with memory management by allocating the appropriate amount of memory for the variable.

1. Differentiate between static, instance and local variables.

**Static Variables**: Also known as class variables, they are declared with the static keyword within a class but outside any method. They are shared among all instances of the class and have only one copy that exists regardless of the number of instances of the class.

**Instance Variables**: These are non-static variables and are declared within a class but outside any method. Each instance of the class has its own copy of an instance variable.

**Local Variables**: Declared within a method or block, they are only accessible within that method or block and cease to exist once the method completes execution.

1. Differentiate between widening and narrowing casting in java.

**Widening Casting (Implicit)**: This occurs when data is converted from a smaller to a larger data type, like converting an int to a long.

**Narrowing Casting (Explicit)**: This happens when data is converted from a larger to a smaller data type, like converting a double to an int. This requires explicit casting and can lead to loss of information if the value does not fit within the range of the target type.

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 | true | true, false |
| Char | 2 | 0 | ‘\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 | ±3.40282347E+38F to  to ±1.4E-45F |
| Double | 8 | 0.0d | -1.8E+308 to +1.8E+308 |

1. Define class as used in OOP

In Object-Oriented Programming (OOP), a **class** is a blueprint or template for creating objects. It defines a set of properties (attributes or fields) and methods (functions or behaviors) that the objects created from the class will have

1. Explain the importance of classes in Java programming.

**Encapsulation**: Classes bundle data and methods that operate on the data into a single unit.

**Code Reusability**: Classes allow you to create objects as instances, enabling you to reuse code without redundancy.

**Inheritance**: Classes can inherit properties and methods from other classes, promoting code reusability and logical hierarchy.

**Abstraction**: Classes provide abstraction by allowing you to create complex data types that represent real-world entities without exposing unnecessary implementation details.

**Modularity**: Classes help in organizing code into different modules, which makes the program easier to manage and scale.

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

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