## Batch - CMJD106

## Module – Programming Fundamentals

## Assignment - 03

[01]

import java.util.Scanner;

class Example{

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int num1 = scanner.nextInt();

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

int num2 = scanner.nextInt();

int sum = num1 + num2;

System.out.println("Sum: " + sum);

}

}

[02]

import java.util.Scanner;

class InputOutput {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter first value: ");

int value1 = scanner.nextInt()

System.out.print("Enter second value: ");

int value2 = scanner.nextInt();

System.out.println("Values are " + value1 + " and " + value2);

}

}

[03]

import java.util.Scanner;

class AgeAfterThreeYears {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int age = scanner.nextInt();

age += 3;

System.out.println("New age: " + age);

}

}

[04]

* 60
* 10+20+30
* 10+2030
* 102030
* 102030
* 3030
* 102030

[05]

a,c,d

[06]

* a.6, Addition of integers.
* b. 123, Concatenation of strings.
* c. 150, Unicode values of characters are added.
* d. System.1 2 3, Characters are concatenated as strings.
* e. 198Unicode values of characters are added.
* f. ABC, Concatenation of strings.
* g. 415, Character 'A' is converted to its Unicode value and then added.
* h. A B C, Characters are concatenated as strings.

[07]

* + true
  + true
  + true
  + true
  + true
  + false
  + true

[08]

age += 10;

[09]

num1 = num1 + num2;

num2 = num1 - num2;

num1 = num1 - num2;

[11]

B, E

[12]

a, c, e

[13]

avg = (double) tot / 10;

[14]

D

[15]

a. char a = '\u0061';

[16]

B,d

[17]

C

[18]

* 3
* -3
* 3
* -3
* 3
* -3
* 3

[19]

Line 1

[20]

A, c, e

[21]

A, c

[22]

* 17
* -10
* -17
* 3
* 7
* -3

[23]

* 100
* -100
* 100
* 200
* -400
* 0

[24]

* 100
* 101
* 103
* 103

[25]

* 101 100
* 102 101
* 103 102

[26]

* 101 101
* 102 102
* 103 103

[27]

* 100
* 100
* 100
* 101
* 101
* 102
* 102
* 103
* 103

[28]

* 3
* 0
* 10
* 0.0
* 0.5

[30]

12 - 4 \* 2 : 4

(12 - 4) \* 2 : 16

12 - (4 \* 2) : 4

[31]

* x = 7 % 10 / 2 \* 2;: This evaluates to 0. Modulus (%) has higher precedence than division (/), so 7 % 10 is calculated first, which is 7. Then, division and multiplication are performed from left to right.
* b. x = 7 % (10 / 2) \* 2;: This evaluates to 2. Parentheses have the highest precedence, so 10 / 2 is calculated first, which is 5. Then, modulus (%) and multiplication are performed from left to right.
* c. x = 7 % 10 / (2 \* 2);: This evaluates to 0. Parentheses have the highest precedence, so 2 \* 2 is calculated first, which is 4. Then, modulus (%) and division (/) are performed from left to right.
* d. x = 7 % (10 / (2 \* 2));: This evaluates to 1. Parentheses have the highest precedence, so 2 \* 2 is calculated first, which is 4. Then, division (/) and modulus (%) are performed from left to right.
* e. x = 7 % ((10 / 2) \* 2);: This evaluates to 1. Parentheses have the highest precedence, so 10 / 2 is calculated first, which is 5. Then, multiplication and modulus (%) are performed from left to right.

[32]

* a = a + (a = 6);: This expression is undefined behavior in Java. It tries to modify a and access its value simultaneously, leading to unpredictability and violating the sequence point concept in Java.
* a = (a = 6) + a;: This expression assigns 6 to a, then adds the value of a (which is 6) to itself and assigns the result back to a. So, a becomes 12.
* a = (a = 6) + (a = 5);: This expression is also undefined behavior. It attempts to modify a and access its value simultaneously, leading to unpredictability.
* a = a \* 3 + a;: This expression multiplies the current value of a by 3, adds the result to the current value of a, and assigns the final result back to a.