

## 4-Stage Pipeline Performance Measurement

EXP NO: 36

AIM:

To write a C program to simulate a 4-stage pipeline for basic arithmetic operations and measure its performance.

ALGORITHM:

1. **Start**
2. **Initialize** variables:
  - counter to track the number of cycles.
  - input, num1, num2, op, res, ins, and performance\_measure for various inputs and calculations.
3. **Prompt the user** for the first operand (num1), second operand (num2), and the operation to perform.
  - Increment counter for each input operation.
4. **Perform the selected operation** based on user input:
  - **Addition:**  $res = num1 + num2$
  - **Subtraction:**  $res = num1 - num2$
  - **Multiplication:**  $res = num1 * num2$
  - **Division:** Check if num2 is zero; otherwise, perform  $res = num1 / num2$
  - Increment counter after the operation.
5. **Handle invalid cases** by incrementing the counter by 3.
6. **Display the number of cycles** used.
7. **Prompt the user** for the number of instructions (ins).
8. **Calculate the performance measure** as  $ins / counter$ .
9. **Display the performance measure.**
10. **End**

PROGRAM:

```
#include <stdio.h>
```

```
void main() {
```

```
    int counter = 0;
```

```
    int num1, num2;
```

```
    int op;
```

```
    int res;
```

```
    int ins;
```

```
    int performance_measure = 0;
```

```
// Input first value
printf("\nEnter 1st value: ");
scanf("%d", &num1);
counter += 1;

// Input second value
printf("\nEnter the 2nd value: ");
scanf("%d", &num2);
counter += 1;

// Input operation choice
printf("\nEnter the option:\n1) Addition\n2) Subtraction\n3) Multiplication\n4) Division");
scanf("%d", &op);

switch(op) {
    case 1:
        printf("Performing addition operation");
        res = num1 + num2;
        counter += 1;
        break;
    case 2:
        printf("Performing subtraction operation");
        res = num1 - num2;
        counter += 1;
        break;
    case 3:
        printf("Performing multiplication operation");
        res = num1 * num2;
        counter += 1;
        break;
    case 4:
        if (num2 == 0) {
            printf("\nDenominator can't be zero");
        } else {
```

```
        printf("Performing division operation");

        res = num1 / num2;

        counter += 1;
    }

    break;

default:

    printf("Invalid case...");

    counter += 3;

    break;
}
```

```
printf("\nCYCLE VALUE IS: %d", counter);
```

```
// Input number of instructions
```

```
printf("\nEnter the number of instructions: ");
```

```
scanf("%d", &ins);
```

```
// Calculate performance measure
```

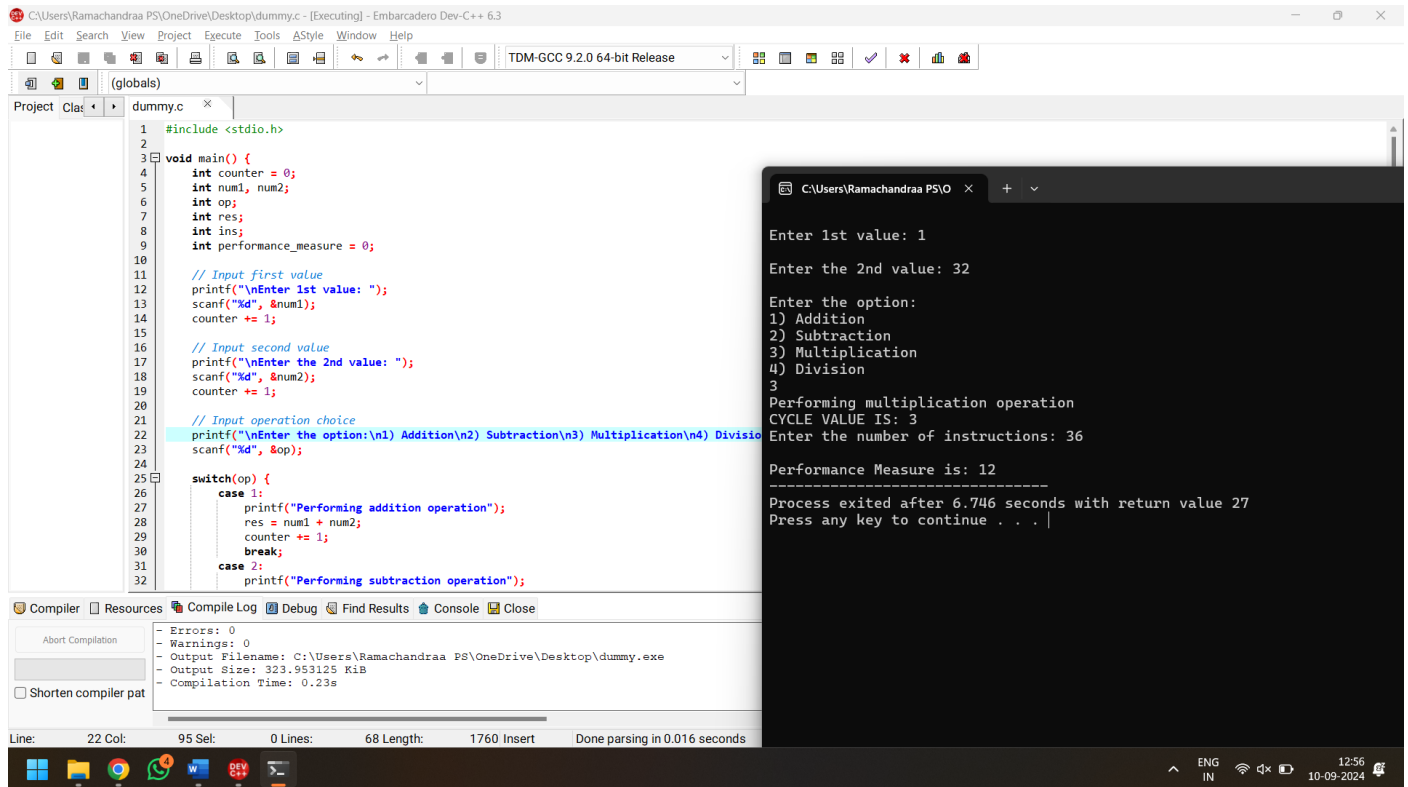
```
performance_measure = ins / counter;
```

```
// Display performance measure
```

```
printf("\nPerformance Measure is: %d", performance_measure);
```

```
}
```

## INPUT & OUTPUT:



The screenshot displays the DevC++ IDE with a C program named 'dummy.c' open. The program is a simple calculator that takes two numbers and an operation as input, performs the operation, and outputs the result along with performance metrics. The code is as follows:

```
1 #include <stdio.h>
2
3 void main() {
4     int counter = 0;
5     int num1, num2;
6     int op;
7     int res;
8     int ins;
9     int performance_measure = 0;
10
11     // Input first value
12     printf("\nEnter 1st value: ");
13     scanf("%d", &num1);
14     counter += 1;
15
16     // Input second value
17     printf("\nEnter the 2nd value: ");
18     scanf("%d", &num2);
19     counter += 1;
20
21     // Input operation choice
22     printf("\nEnter the option:\n1) Addition\n2) Subtraction\n3) Multiplication\n4) Division\n");
23     scanf("%d", &op);
24
25     switch(op) {
26     case 1:
27         printf("Performing addition operation");
28         res = num1 + num2;
29         counter += 1;
30         break;
31     case 2:
32         printf("Performing subtraction operation");
```

The output window shows the program's execution with the following text:

```
Enter 1st value: 1
Enter the 2nd value: 32
Enter the option:
1) Addition
2) Subtraction
3) Multiplication
4) Division
3
Performing multiplication operation
CYCLE VALUE IS: 3
Enter the number of instructions: 36
Performance Measure is: 12
-----
Process exited after 6.746 seconds with return value 27
Press any key to continue . . . |
```

The bottom status bar of the IDE indicates the current line is 22, column is 22, and the file is 95 lines long. The compilation time is 0.23s.

**RESULT:** Thus, the program was executed successfully using DevC++