

CPU PERFORMANCE

EXP NO: 26

AIM: To write a C program to implement CPU performance measures.

ALGORITHM:

Step 1: start

Step 2: Declare the necessary variables: cr (clock rate), p (number of processors), p1 (a copy of the number of processors), i (loop variable), and cpu (array to store CPU times).

Step 3: Initialize the cpu array elements to 0.

Step 4: Prompt the user to enter the number of processors (p).

Step 5: Store the value of p in p1.

Step 6: Start a loop from 0 to p-1:

- a. Prompt the user to enter the cycles per instruction (cpi) for the current processor.
- b. Prompt the user to enter the clock rate (cr) in GHz for the current processor.
- c. Calculate the CPU time (ct) using the formula: $ct = 1000 * cpi / cr$.
- d. Display the CPU time for the current processor.
- e. Store the CPU time in the cpu array at index i.

Step 7: Set max as the first element of the cpu array.

Step 8: Start a loop from 0 to p1-1:

a. If the CPU time at index i is less than or equal to max, update max to the current CPU time.

Step 9: Display the processor with the lowest execution time (max).

Step 10: Exit the program.

PROGRAM:

```
#include <stdio.h>

int main()
{
    float cr;
    int p,p1,i;
    float cpu[5];
    float cpi,ct,max;
    int n=1000;
    for(i=0;i<=4;i++)
    {
        cpu[5]=0;
    }
    printf("\n Enter the number of processors:");
    scanf("%d",&p);
    p1=p;
    for(i=0;i<p;i++)
    {
        printf("\n Enter the Cycles per Instrcution of processor:");
        scanf("%f",&cpi);
        printf("\n Enter the clockrate in GHz:");
        scanf("%f",&cr);
        ct=1000*cpi/cr;
        printf("The CPU time is: %f",ct);
        cpu[i]=ct;
    }
    max=cpu[0];
    for(i=0;i<p1;i++)
    {
        if(cpu[i]<=max)
            max=cpu[i];
    }
    printf("\n The processor has lowest Execution time is: %f ", max);
    return 0;
```

}

INPUT:

3

1.0

2.0

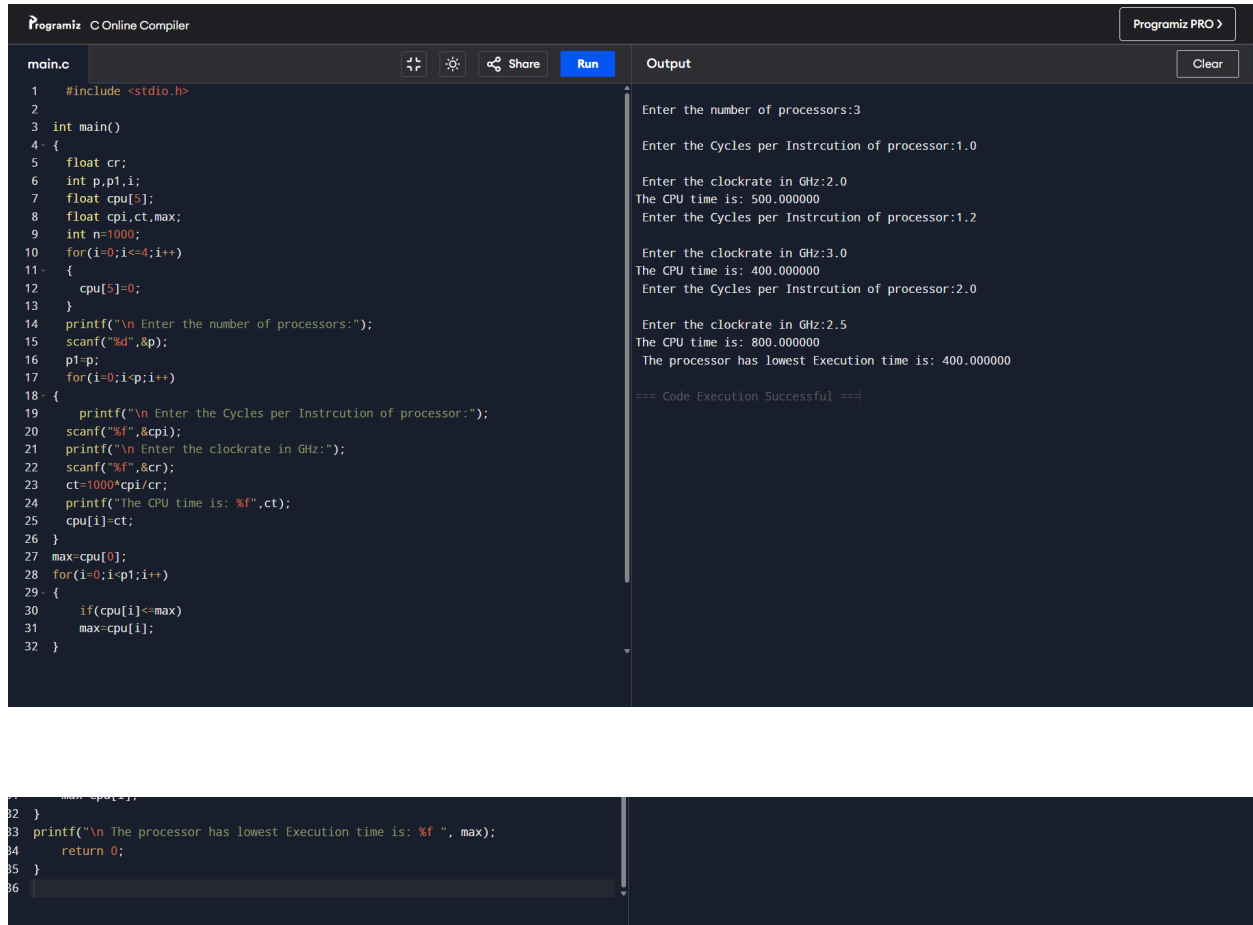
1.2

3.0

2.0

2.5

OUTPUT:



The screenshot displays the Programiz C Online Compiler interface. The left pane shows the source code in `main.c`, and the right pane shows the program's output.

Source Code (main.c):

```
1  #include <stdio.h>
2
3  int main()
4  {
5      float cr;
6      int p,p1,i;
7      float cpu[S];
8      float cpi,ct,max;
9      int n=1000;
10     for(i=0;i<=4;i++)
11     {
12         cpu[i]=0;
13     }
14     printf("\n Enter the number of processors:");
15     scanf("%d",&p);
16     p1=p;
17     for(i=0;i<p;i++)
18     {
19         printf("\n Enter the Cycles per Instruction of processor:");
20         scanf("%f",&cpi);
21         printf("\n Enter the clockrate in GHz:");
22         scanf("%f",&cr);
23         ct=1000*cpi/cr;
24         printf("The CPU time is: %f",ct);
25         cpu[i]=ct;
26     }
27     max=cpu[0];
28     for(i=0;i<p1;i++)
29     {
30         if(cpu[i]<=max)
31             max=cpu[i];
32     }
```

Output:

```
Enter the number of processors:3
Enter the Cycles per Instruction of processor:1.0
Enter the clockrate in GHz:2.0
The CPU time is: 500.000000
Enter the Cycles per Instruction of processor:1.2
Enter the clockrate in GHz:3.0
The CPU time is: 400.000000
Enter the Cycles per Instruction of processor:2.0
Enter the clockrate in GHz:2.5
The CPU time is: 800.000000
The processor has lowest Execution time is: 400.000000

=== Code Execution Successful ===
```

The bottom portion of the image shows a continuation of the code editor, displaying lines 32 through 36 of the program, which include the final loop for finding the maximum execution time and the return statement.

RESULT: Thus the program was executed successfully using DevC++.

