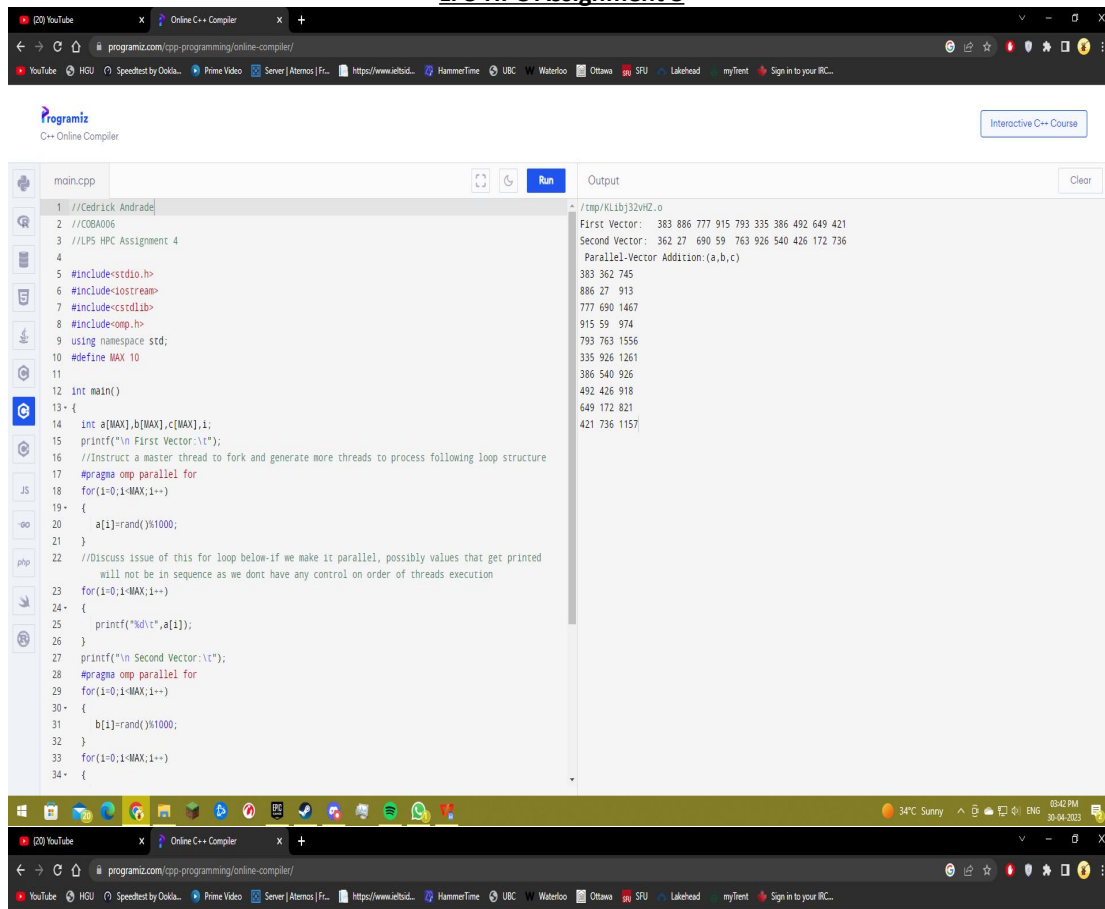


## LP5 HPC Assignment 5

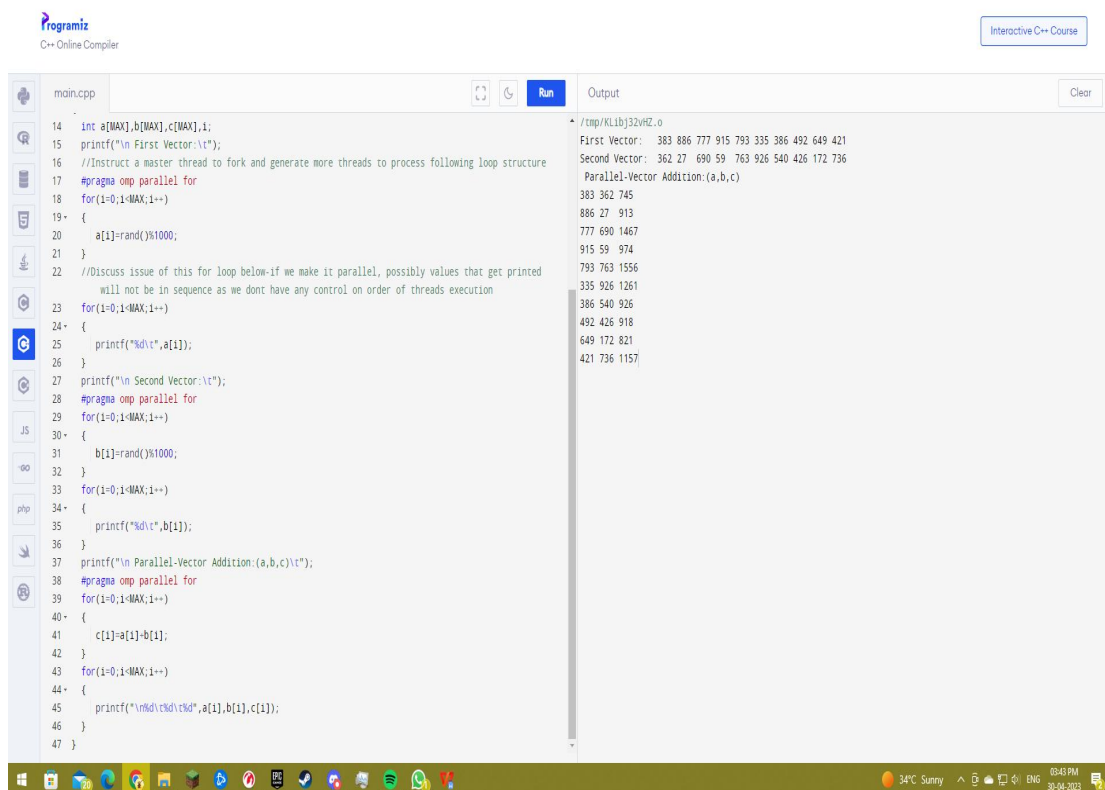


The screenshot shows the Programiz C++ Online Compiler interface. The code in `main.cpp` is as follows:

```
1 //Cedrick Andrade
2 //COBA006
3 //LP5 HPC Assignment 4
4
5 #include<stdio.h>
6 #include<iostream>
7 #include<stdlib.h>
8 #include<omp.h>
9 using namespace std;
10 #define MAX 10
11
12 int main()
13 {
14     int a[MAX],b[MAX],c[MAX],i;
15     printf("\n First Vector:\t");
16     //Instruct a master thread to fork and generate more threads to process following loop structure
17     #pragma omp parallel for
18     for(i=0;i<MAX;i++)
19     {
20         a[i]=rand()%1000;
21     }
22     //Discuss issue of this for loop below-if we make it parallel, possibly values that get printed
23     //will not be in sequence as we dont have any control on order of threads execution
24     for(i=0;i<MAX;i++)
25     {
26         printf("%d\t",a[i]);
27     }
28     printf("\n Second Vector:\t");
29     #pragma omp parallel for
30     for(i=0;i<MAX;i++)
31     {
32         b[i]=rand()%1000;
33     }
34     for(i=0;i<MAX;i++)
```

The output shows the results of the parallel execution:

```
/tmp/KLibj32vH2.o
First Vector: 383 886 777 915 793 335 386 492 649 421
Second Vector: 362 27 690 59 763 926 540 426 172 736
Parallel-Vector Addition:(a,b,c)
383 362 745
886 27 913
777 690 1467
915 59 974
793 763 1556
335 926 1261
386 540 926
492 426 918
649 172 821
421 736 1157
```



The screenshot shows the Programiz C++ Online Compiler interface. The code in `main.cpp` is as follows:

```
14 int a[MAX],b[MAX],c[MAX],i;
15 printf("\n First Vector:\t");
16 //Instruct a master thread to fork and generate more threads to process following loop structure
17 #pragma omp parallel for
18 for(i=0;i<MAX;i++)
19 {
20     a[i]=rand()%1000;
21 }
22 //Discuss issue of this for loop below-if we make it parallel, possibly values that get printed
23 //will not be in sequence as we dont have any control on order of threads execution
24 for(i=0;i<MAX;i++)
25 {
26     printf("%d\t",a[i]);
27 }
28 printf("\n Second Vector:\t");
29 #pragma omp parallel for
30 for(i=0;i<MAX;i++)
31 {
32     b[i]=rand()%1000;
33 }
34 for(i=0;i<MAX;i++)
35 {
36     printf("%d\t",b[i]);
37 }
38 printf("\n Parallel-Vector Addition:(a,b,c)\t");
39 #pragma omp parallel for
40 for(i=0;i<MAX;i++)
41 {
42     c[i]=a[i]+b[i];
43 }
44 for(i=0;i<MAX;i++)
45 {
46     printf("\n%d\t%d\t%d\t",a[i],b[i],c[i]);
47 }
```

The output shows the results of the parallel execution:

```
/tmp/KLibj32vH2.o
First Vector: 383 886 777 915 793 335 386 492 649 421
Second Vector: 362 27 690 59 763 926 540 426 172 736
Parallel-Vector Addition:(a,b,c)
383 362 745
886 27 913
777 690 1467
915 59 974
793 763 1556
335 926 1261
386 540 926
492 426 918
649 172 821
421 736 1157
```