

Practical No 4

PRN: 22520005

Name: Aftab Imtiyaj Bhadgaonkar

Batch: B6

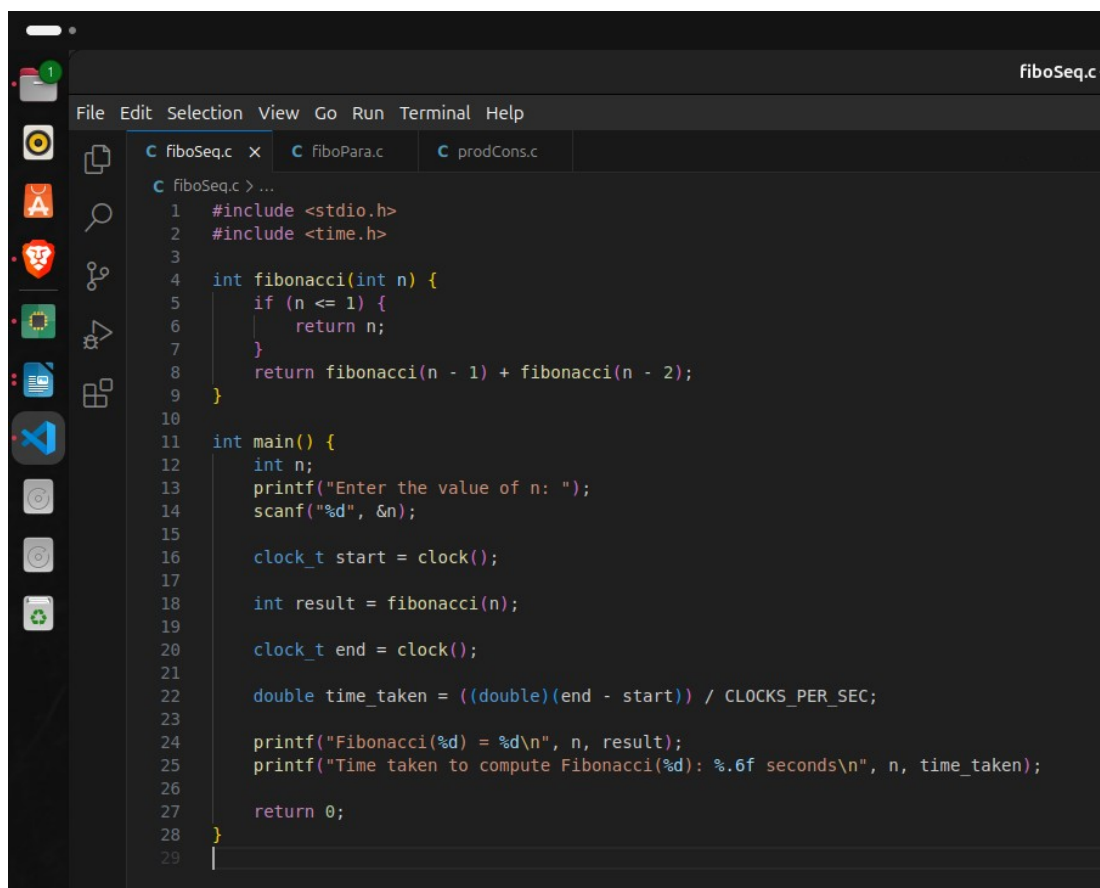
Course: High Performance Computing Lab

Problem Statement 1:

Analyse and implement a Parallel code for below programs using OpenMP considering synchronization requirements. (Demonstrate the use of different clauses and constructs wherever applicable)

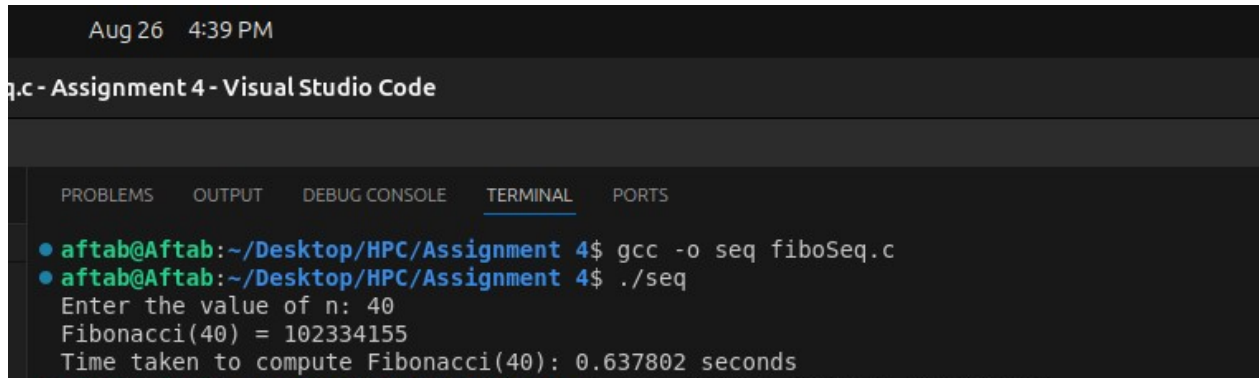
Fibonacci Computation:

Code(Sequential):

A screenshot of a C code editor window titled 'fibonacci.c'. The editor has a menu bar with 'File', 'Edit', 'Selection', 'View', 'Go', 'Run', 'Terminal', and 'Help'. Below the menu bar are three tabs: 'C fibonacci.c', 'C fiboPara.c', and 'C prodCons.c'. The 'C fibonacci.c' tab is active, showing the following C code:

```
1  #include <stdio.h>
2  #include <time.h>
3
4  int fibonacci(int n) {
5      if (n <= 1) {
6          return n;
7      }
8      return fibonacci(n - 1) + fibonacci(n - 2);
9  }
10
11 int main() {
12     int n;
13     printf("Enter the value of n: ");
14     scanf("%d", &n);
15
16     clock_t start = clock();
17
18     int result = fibonacci(n);
19
20     clock_t end = clock();
21
22     double time_taken = ((double)(end - start)) / CLOCKS_PER_SEC;
23
24     printf("Fibonacci(%d) = %d\n", n, result);
25     printf("Time taken to compute Fibonacci(%d): %.6f seconds\n", n, time_taken);
26
27     return 0;
28 }
29
```

Screenshot:



The screenshot shows a terminal window titled "q.c - Assignment 4 - Visual Studio Code". The terminal output is as follows:

```
Aug 26 4:39 PM
q.c - Assignment 4 - Visual Studio Code

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

• aftar@Aftar:~/Desktop/HPC/Assignment 4$ gcc -o seq fiboSeq.c
• aftar@Aftar:~/Desktop/HPC/Assignment 4$ ./seq
Enter the value of n: 40
Fibonacci(40) = 102334155
Time taken to compute Fibonacci(40): 0.637802 seconds
```

Code(Parallel):

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

unsigned long long fibonacci(int n) {
if (n <= 1) return n;

unsigned long long fib1, fib2;

if (n > 5)
{
#pragma omp task shared(fib1)
fib1 = fibonacci(n - 1);

#pragma omp task shared(fib2)
fib2 = fibonacci(n - 2);

#pragma omp taskwait
}
else
{
fib1 = fibonacci(n - 1);
fib2 = fibonacci(n - 2);
}

return fib1 + fib2;
}
```

```
int main() {
int n, num_threads;
unsigned long long result;

printf("Enter the value of n: ");
scanf("%d", &n);

printf("Enter the number of threads : ");
scanf("%d", &num_threads);

omp_set_num_threads(num_threads);

double start_time = omp_get_wtime();

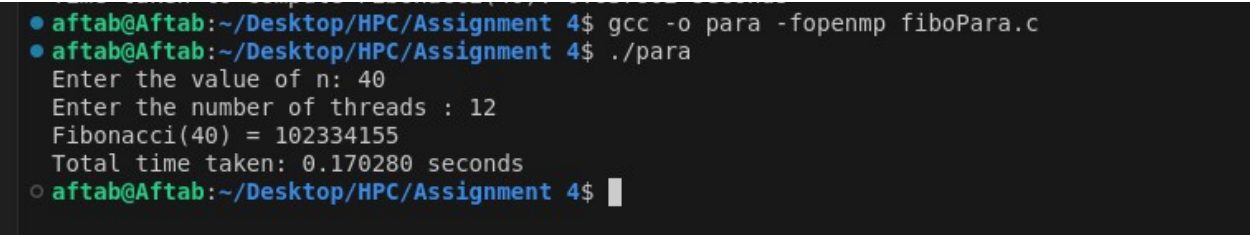
#pragma omp parallel
{
#pragma omp single
{
result = fibonacci(n);
}
}

double end_time = omp_get_wtime();

printf("Fibonacci(%d) = %llu\n", n, result);
printf("Total time taken: %f seconds\n", end_time - start_time);

return 0;
}
```

Screenshots:



```
● aftar@Aftar:~/Desktop/HPC/Assignment 4$ gcc -o para -fopenmp fiboPara.c
● aftar@Aftar:~/Desktop/HPC/Assignment 4$ ./para
Enter the value of n: 40
Enter the number of threads : 12
Fibonacci(40) = 102334155
Total time taken: 0.170280 seconds
○ aftar@Aftar:~/Desktop/HPC/Assignment 4$
```

Analysis:

Input Value	Parallel	Sequential
40	0.170280	0.637802

Problem Statement 2:

Analyse and implement a Parallel code for below programs using OpenMP considering synchronization requirements. (Demonstrate the use of different clauses and constructs wherever applicable)

Producer Consumer Problem

Code:

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

int full = 0;
int empty = 10, x = 0;
omp_lock_t lock;

void producer() {
    omp_set_lock(&lock);
    if (empty > 0) {
        full++;
        empty--;
        x++;
        printf("\nProducer produces item %d\n", x);
    } else {
        printf("\nBuffer is full!\n");
    }
    omp_unset_lock(&lock);
}

void consumer() {
    omp_set_lock(&lock);
    if (full > 0) {
        full--;
        empty++;
        printf("\nConsumer consumes item %d\n", x);
        x--;
    } else {
```

```
printf("\nBuffer is empty!\n");  
}  
omp_unset_lock(&lock);  
}
```

```
int main() {  
int n;
```

```
omp_init_lock(&lock);
```

```
while (1) {  
printf(  
"\n1. Press 1 for Producer"  
"\n2. Press 2 for Consumer"  
"\n3. Press 3 for Exit");
```

```
printf("\nEnter your choice: ");  
scanf("%d", &n);
```

```
switch (n) {  
case 1:  
#pragma omp task  
{  
producer();  
}  
break;
```

```
case 2:  
#pragma omp task  
{  
consumer();  
}  
break;
```

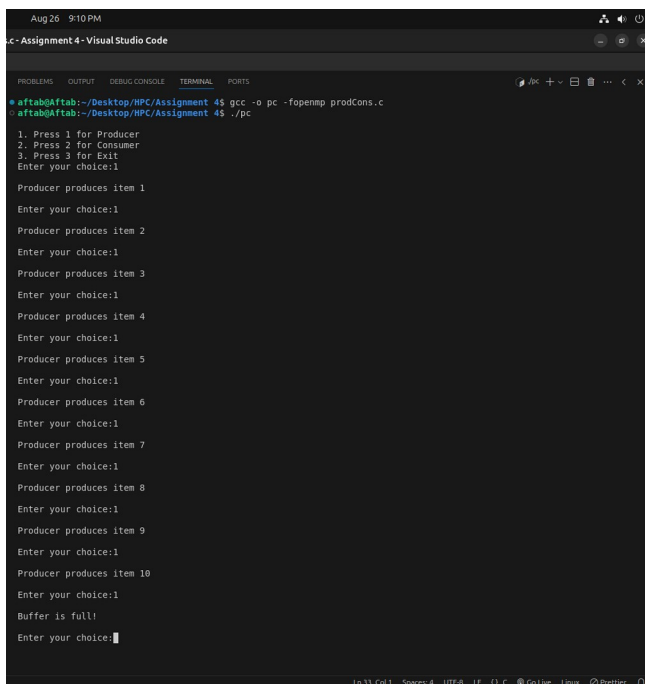
```
case 3:  
omp_destroy_lock(&lock);  
exit(0);
```

```
default:
```

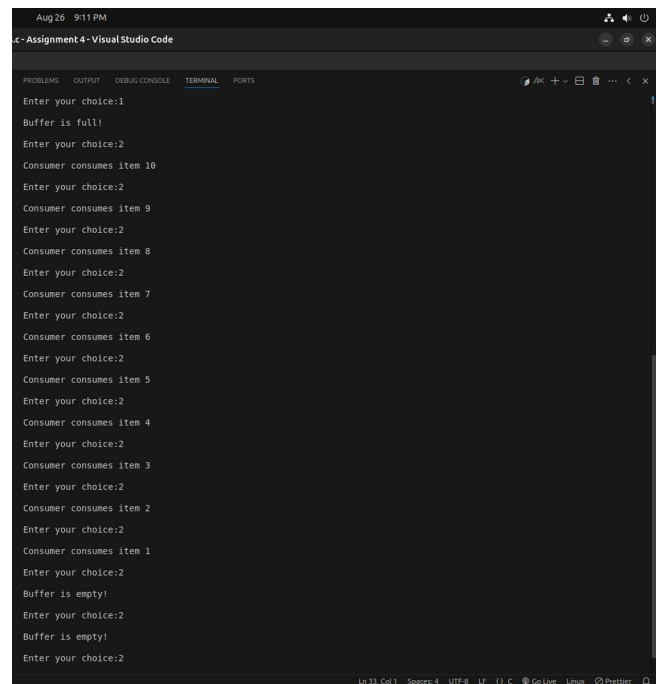
Walchand College of Engineering, Sangli
Department of Computer Science and Engineering

```
printf("\nInvalid choice! Please try again.");  
break;  
}  
}  
  
return 0;  
}
```

Screenshot:



```
Aug 26 9:10 PM  
c- Assignment 4 - Visual Studio Code  
aftab@aftab:~/Desktop/NPC/Assignment 4$ gcc -o pc -fopenmp prodCons.c  
aftab@aftab:~/Desktop/NPC/Assignment 4$ ./pc  
1. Press 1 for Producer  
2. Press 2 for Consumer  
3. Press 3 for Exit  
Enter your choice:1  
Producer produces item 1  
Enter your choice:1  
Producer produces item 2  
Enter your choice:1  
Producer produces item 3  
Enter your choice:1  
Producer produces item 4  
Enter your choice:1  
Producer produces item 5  
Enter your choice:1  
Producer produces item 6  
Enter your choice:1  
Producer produces item 7  
Enter your choice:1  
Producer produces item 8  
Enter your choice:1  
Producer produces item 9  
Enter your choice:1  
Producer produces item 10  
Enter your choice:1  
Buffer is full!  
Enter your choice:1
```



```
Aug 26 9:11 PM  
c- Assignment 4 - Visual Studio Code  
Enter your choice:1  
Buffer is full!  
Enter your choice:2  
Consumer consumes item 10  
Enter your choice:2  
Consumer consumes item 9  
Enter your choice:2  
Consumer consumes item 8  
Enter your choice:2  
Consumer consumes item 7  
Enter your choice:2  
Consumer consumes item 6  
Enter your choice:2  
Consumer consumes item 5  
Enter your choice:2  
Consumer consumes item 4  
Enter your choice:2  
Consumer consumes item 3  
Enter your choice:2  
Consumer consumes item 2  
Enter your choice:2  
Consumer consumes item 1  
Enter your choice:2  
Buffer is empty!  
Enter your choice:2  
Buffer is empty!  
Enter your choice:2
```