University of Wolverhampton
School of Mathematics and Computer Science

Student Id:2052261
Name: MahanTimalsena

6CS005 High Performance Computing Week 1 Workshop

Revision on C and Multithreading

Tasks – Basic C Syntax

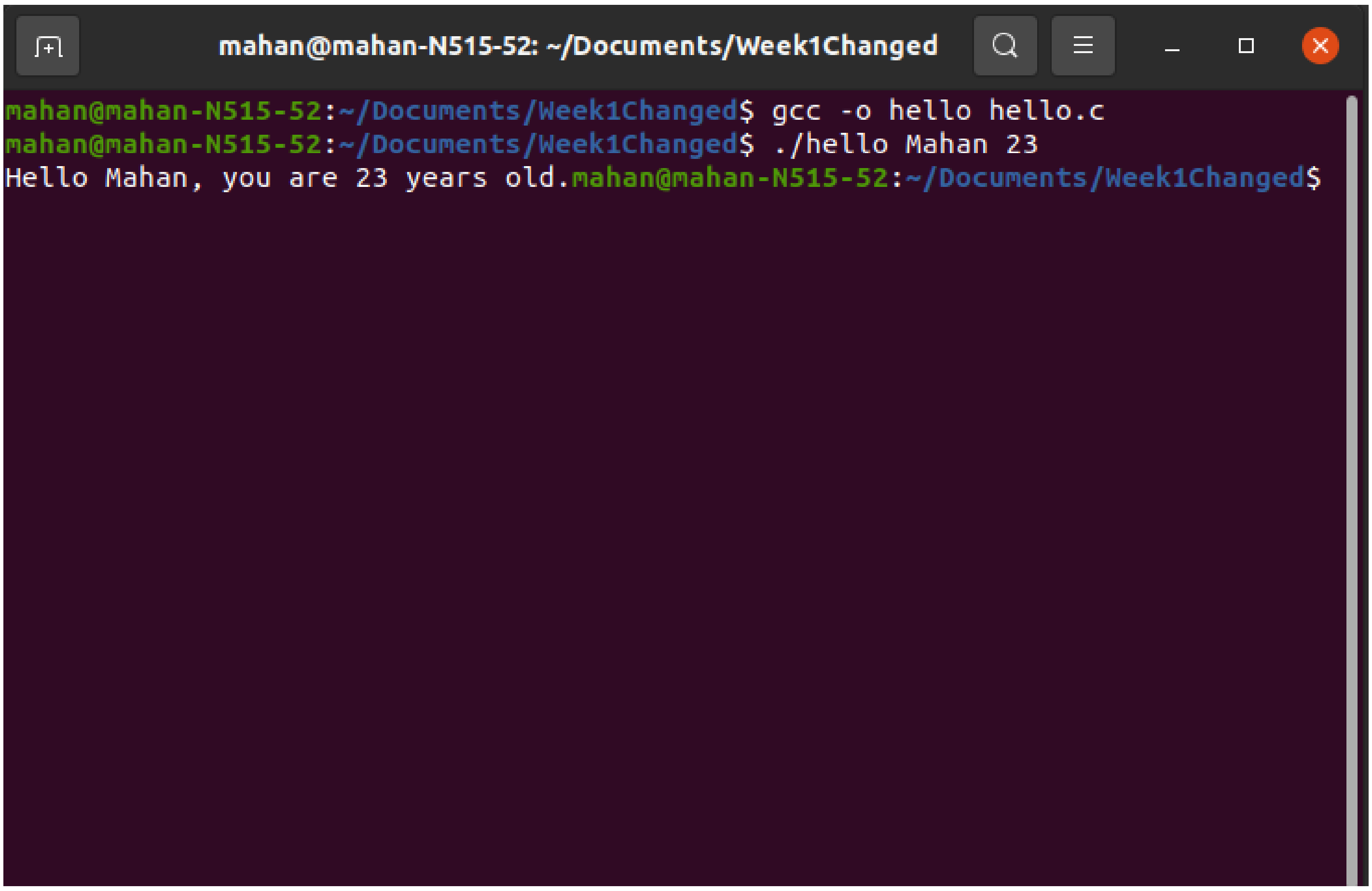
The following code prints out the value of an `int` variable and a string (`char *`) :

```
#include <stdio.h>

void main(int argc, char *argv[])
{
  int age = 10;   char
*name = "Hiran";
  printf("Hello %s, you are %d years old.", name, age);
}
```

1. Now modify the program so that it uses the command line arguments to supply name and age. i.e. it uses the *argc* and *argv* arguments/parameters.

Code:

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

void main(int argc, char* argv[])
{
int age = atoi(argv[2]);
char *name = argv[1];
printf("Hello %s, you are %d years old.", name, age);
}
```

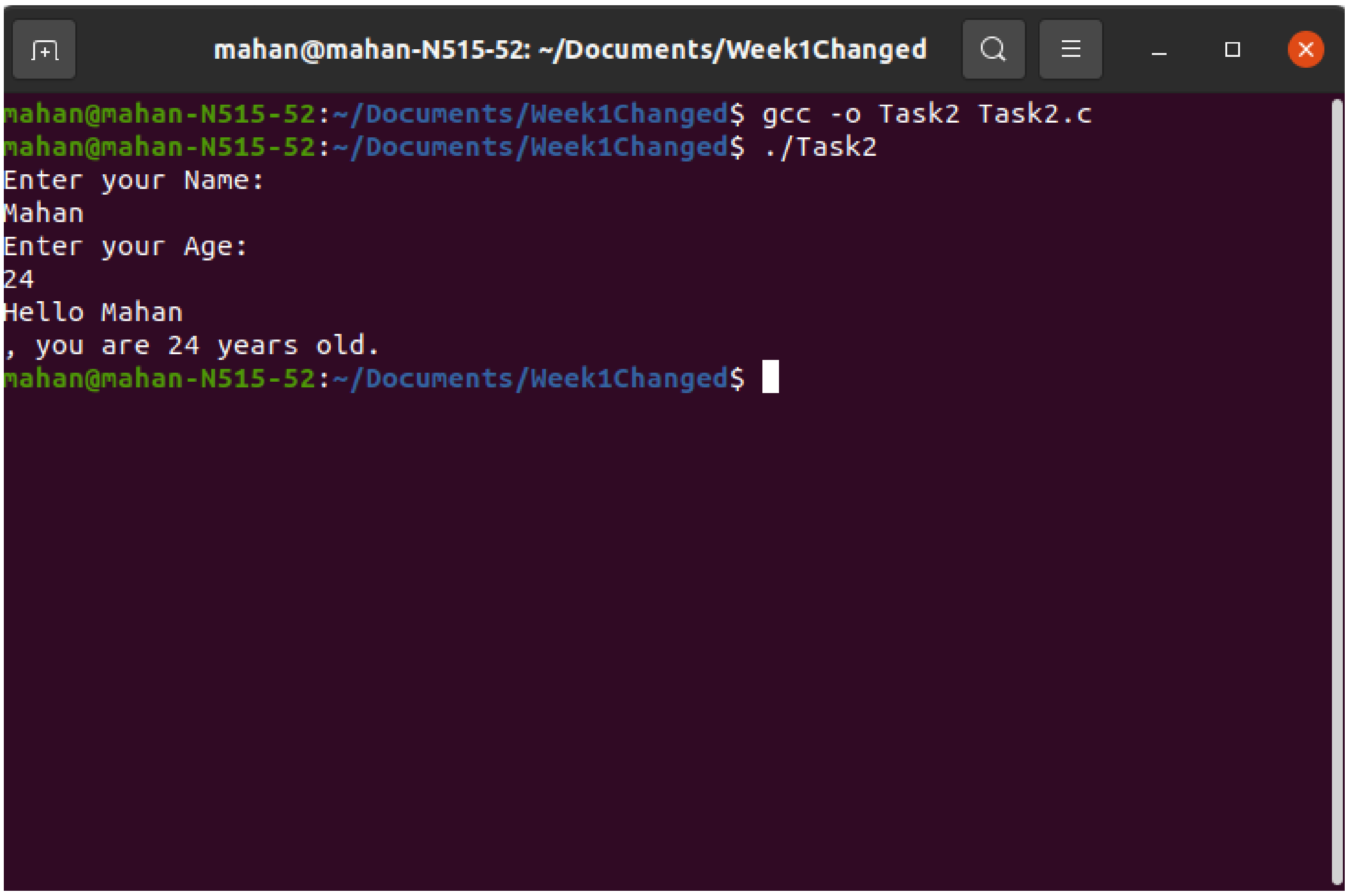


```
mahan@mahan-N515-52:~/Documents/Week1Changed$ gcc -o hello hello.c
mahan@mahan-N515-52:~/Documents/Week1Changed$ ./hello Mahan 23
Hello Mahan, you are 23 years old.mahan@mahan-N515-52:~/Documents/Week1Changed$
```

2. Now modify the program again so that it uses the scanf() function to get input from the user for the name and age.

Code:

```
 #include <stdio.h>
#include <stdint.h>
void main(int argc, char *argv[])
{
int age;
char name[20];
printf("Enter your Name:\n");
fgets(name,20,stdin);
printf("Enter your Age:\n");
scanf("%d",&age);
printf("Hello %s, you are %d years old.\n", name, age);
}
```



```
mahan@mahan-N515-52:~/Documents/Week1Changed$ gcc -o Task2 Task2.c
mahan@mahan-N515-52:~/Documents/Week1Changed$ ./Task2
Enter your Name:
Mahan
Enter your Age:
24
Hello Mahan
, you are 24 years old.
mahan@mahan-N515-52:~/Documents/Week1Changed$ 
```

3) The following code count the integer variable ***n*** from 0 to 9 and prints out “Odd” if ***n*** is even and just the value of ***n*** if it is even:

```
#include <stdio.h>

void main(int argc, char *argv[])
{
  for(int n =0; n <10; n++){
if(n % 2 == 1){
      printf("%d is Odd\n", n);
    }
else{
      printf("%d\n", n);
    }
  }
}
```

When you run the program, it should output the following:

```
0
1 is Odd 2
3 is Odd 4
5 is Odd 6
7 is Odd 8
9 is Odd
```

3. Now modify the program so that it counts the variable n from 1 to 100 and, if n is a multiple of 2 (eg. 2, 4, 6, etc), it would print out the word “Bish”, and if n is a multiple of 3 (eg. 3, 6, 9. 12 etc), it would print out the word “Bash”, and if n is a multiple of 5 (eg. 5, 10, 15 etc), it would print out the word “Bosh”.

However, if n is a multiple of 2 and 3 (eg. 6), it would print out the words “BishBash”, and if n is a multiple of 2 and 5 (eg. 10), it would print out the words “BishBosh”, and if n is a multiple of 3 and 5 (eg. 15), it would print out the words “BashBosh”. Finally, if n is a multiple of 2, 3 and 5 (eg. 30), it would print out the words “BishBashBosh”.

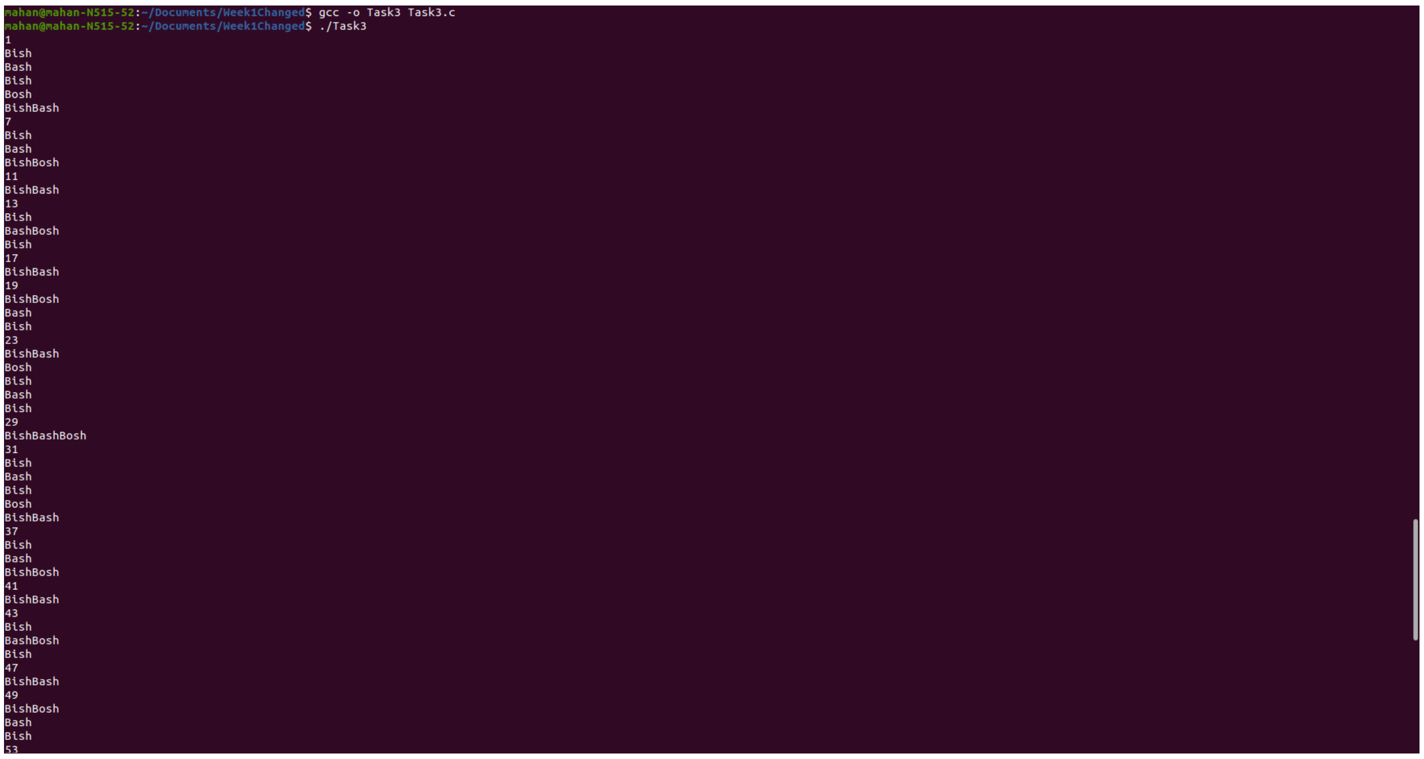
When you run the program, it will produce something like this:

```
1
Bish
Bash
Bish
Bosh
BishBash
7
Bish
Bash
BishBosh 11
BishBash
13
Bish
BashBosh
Bish
17
BishBash 19
BishBosh
Bash
Bish
23
BishBash
Bosh
Bish
Bash
Bish
29
BishBashBosh
31 Bish
Bash
```

Code:

```
 #include <stdio.h>
void main(int argc, char *argv[])
{
        for(int n =1; n <100; n++)
        {
                if(n % 2 == 0)
                {
                        if(n % 3 == 0 )
                        {
                          if(n % 5 == 0 )
                          {
```

```
                printf("BishBashBosh\n");
                            }else
                            {
                                        printf("BishBash\n");
                            }
                        }
                        else if(n % 5 == 0 ){

        printf("BishBosh\n");

                        }
                        else{
                         printf("Bish\n");
                        }

                }
                else if(n % 3 == 0 )
                {
                        if(n % 5 == 0 )
                        {

        printf("BashBosh\n");

                        }else{
        printf("Bash\n");
                        }

                }
                else if(n % 5 == 0 )
                {

        printf("Bosh\n");

                }
                else{
                        printf("%d\n", n);
                }
        }
}
```



```
mahan@mahan-N515-52:~/Documents/Week1Changed$ gcc -o Task3 Task3.c
mahan@mahan-N515-52:~/Documents/Week1Changed$ ./Task3
1
Bish
Bash
Bish
Bosh
BishBash
7
Bish
Bash
BishBosh
11
BishBash
13
Bish
BashBosh
Bish
17
BishBash
19
BishBosh
Bash
Bish
23
BishBash
Bosh
Bish
Bash
Bish
29
BishBashBosh
31
Bish
Bash
Bish
Bosh
BishBash
37
Bish
Bash
BishBosh
41
BishBash
43
Bish
BashBosh
Bish
47
BishBash
49
BishBosh
Bash
Bish
53
```

4) The following code swaps the values of the two variables a and b ::

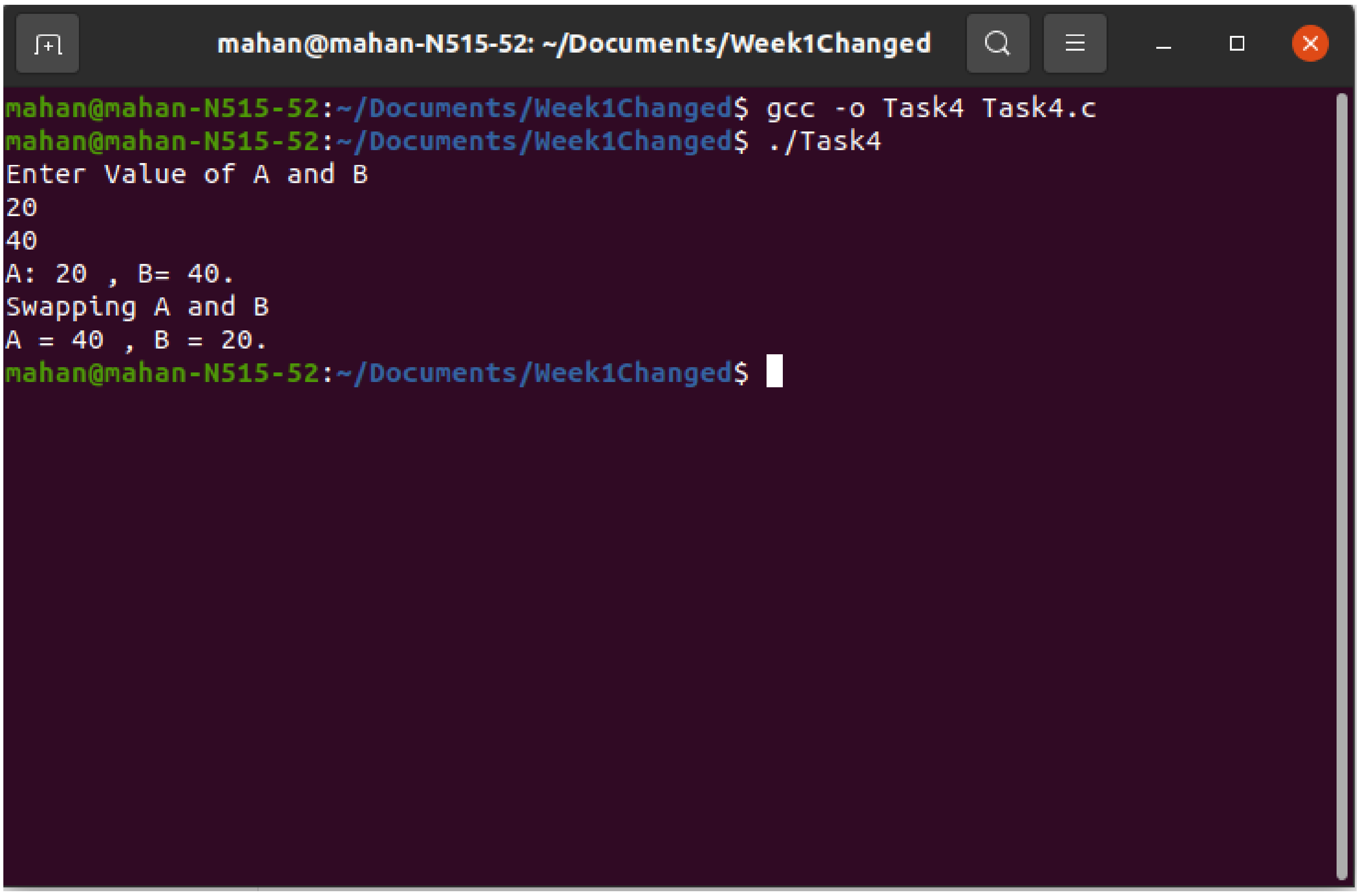
```
#include <stdio.h>

void main(int argc, char *argv[])
{   int a = 3;
int b = 4;
int temp = 0;

  printf("a is %d and b is %d\n", a, b);
     temp =
a;    a = b;
b = temp;

  printf("a is now %d and b is now %d", a, b);
}
```

4. Now write a function called swap() that would swap the values of the variables a and b, when you call the swap() with the variables a and b as parameters. Please note, this exercise requires pointers.

Code:

```
#include<stdio.h>

void swap(int*x, int*y)
{
  int temp;

  temp = *x;
  *x = *y;
  *y = temp;

  }

int main()
{
  int a,b;

  printf("Enter Value of A and B\n");
  scanf("%d %d",&a, &b);

  printf("A: %d , B= %d. \n", a, b);

  printf("Swapping A and B\n");

  swap(&a, &b );

  printf("A = %d , B = %d.\n", a, b);
```

}



```
mahan@mahan-N515-52:~/Documents/Week1Changed$ gcc -o Task4 Task4.c
mahan@mahan-N515-52:~/Documents/Week1Changed$ ./Task4
Enter Value of A and B
20
40
A: 20 , B= 40.
Swapping A and B
A = 40 , B = 20.
mahan@mahan-N515-52:~/Documents/Week1Changed$ 
```

5) The following program fills an int array of size 10 and fills it with random numbers and prints them out:

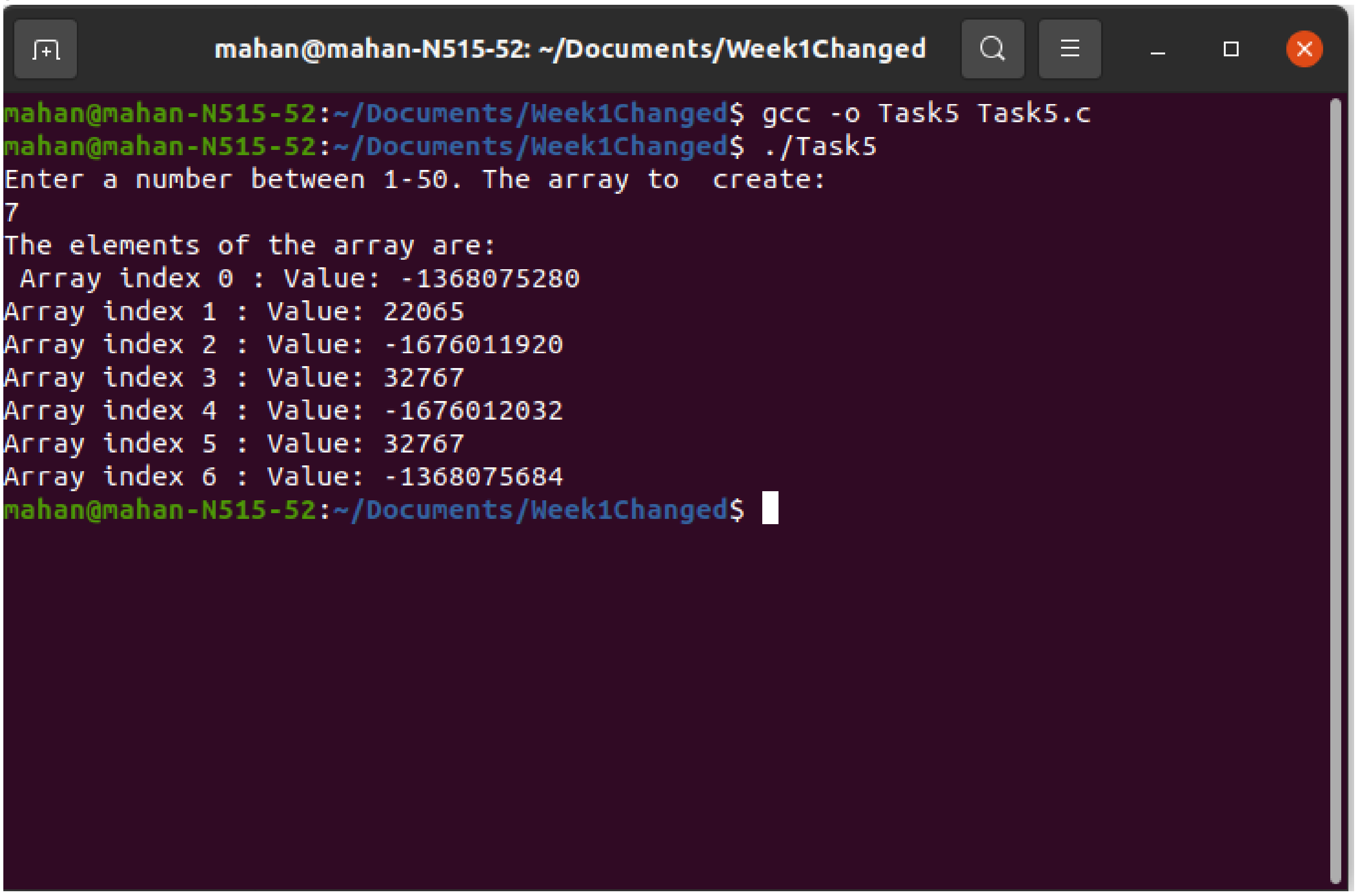
```
#include <stdio.h>
#include <stdlib.h>

void main(int argc, char *argv[])
{
  int numbers[10];

  for (int i=0; i < 10; i++){
numbers[i] = rand();
    printf("%d is %d\n", i, numbers[i]);
  }
}
```

5. Now modify it to will ask the user for a number between 1 and 50, and then use the C function ***`malloc()`*** to allocate an ***`int`*** array of that size, fill it with random numbers and print out the value of each element of that array.

Code:

```
#include <stdio.h>
#include <stdlib.h>
void main(int argc, char *argv[])
{
 int n,i=0;
 int *ptr=0;


 printf("Enter a number between 1-50. The array to  create: \n");
 scanf("%d",&n);
 ptr = (int *)malloc(n*sizeof(int));

 if(n<1){
    printf("Number Must be greater than 0!\n");
    return;
 }
 else if(n>50)
 {
     printf("Number Must be less than 50!\n");
     return;
 }

int numbers[n];

for( i=0; i < n; i++){
numbers[n] = rand();

}
```

```
 printf("The elements of the array are:\n ");
   for(i = 0; i < n; i++)
   {
      printf("Array index %d : Value: %d \n",i, numbers[i]);
   }


}
```



```
mahan@mahan-N515-52:~/Documents/Week1Changed$ gcc -o Task5 Task5.c
mahan@mahan-N515-52:~/Documents/Week1Changed$ ./Task5
Enter a number between 1-50. The array to  create:
7
The elements of the array are:
 Array index 0 : Value: -1368075280
Array index 1 : Value: 22065
Array index 2 : Value: -1676011920
Array index 3 : Value: 32767
Array index 4 : Value: -1676012032
Array index 5 : Value: 32767
Array index 6 : Value: -1368075684
mahan@mahan-N515-52:~/Documents/Week1Changed$ 
```

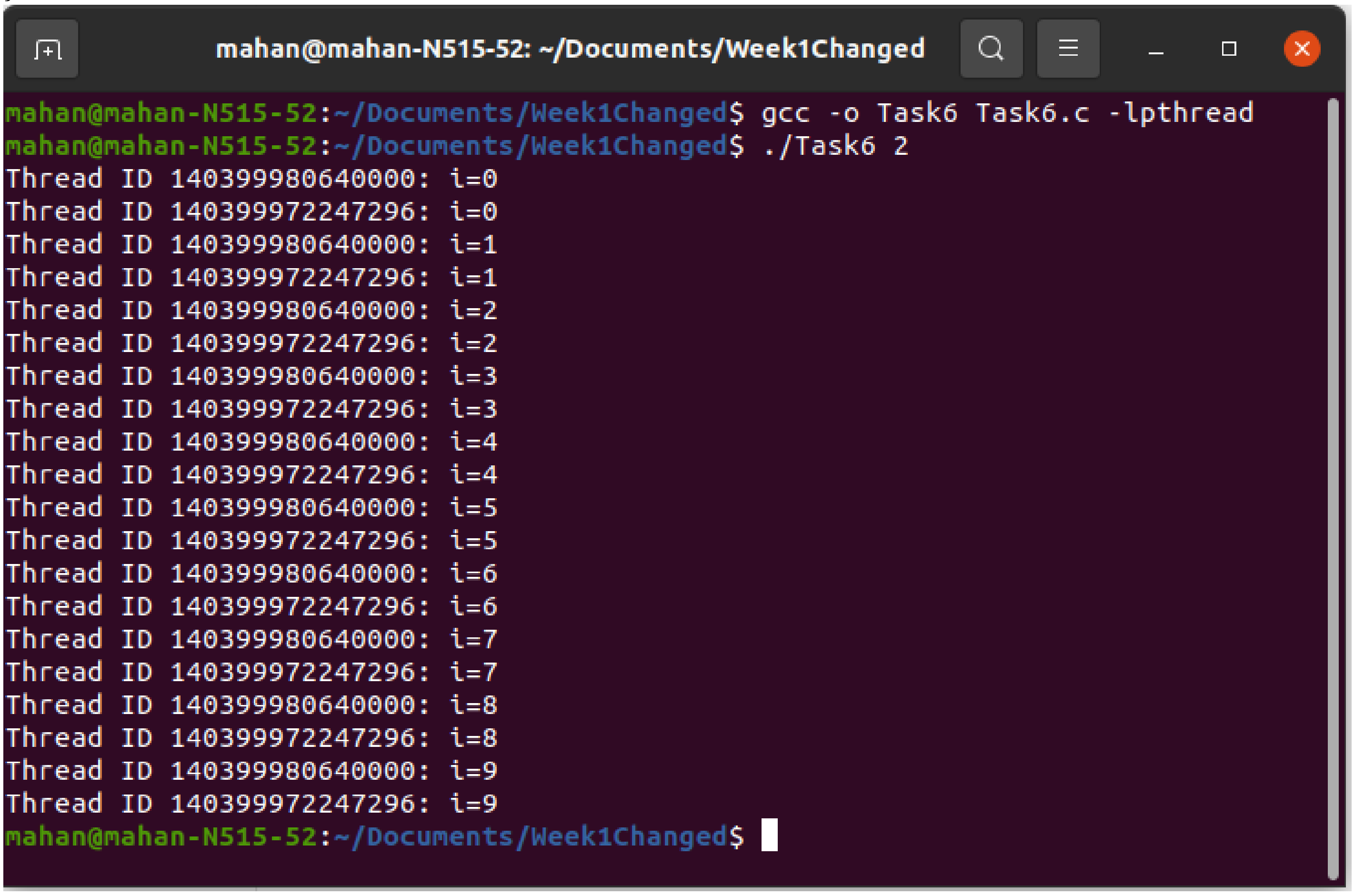
6) The following code creates 2 threads in a program and counts to 10 in each thread :

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

void *threadA(void *p){
for(int i=0; i<10; i++){
    printf("Thread ID %ld: i=%d\n", pthread_self(), i);
usleep(1000);
  }
}
void *threadB(void *p){
for(int i=0; i<10; i++){
    printf("Thread ID %ld: i=%d\n", pthread_self(), i);
usleep(1000);
  }
} void
main(){
  pthread_t thrID1, thrID2;
  pthread_create(&thrID1, NULL, threadA, NULL);
pthread_create(&thrID2, NULL, threadB, NULL);
pthread_join(thrID1, NULL);   pthread_join(thrID2,
NULL);
}
```

6. Modify the program to accept a command line argument to specific the number of threads, and then create that many threads dynamically to run.

Code:

```
#include <pthread.h>
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>

void *thread(void *p){
 for(int i=0; i<10; i++){
 printf("Thread ID %ld: i=%d\n", pthread_self(), i);
 usleep(1000);
 }
}

void main(int argc, char *argv[]){

 int n;
 n  = atoi(argv[1]);
 pthread_t thrId[n];
 for(int i=0; i<n; i++){
pthread_create(&thrId[i], NULL, thread, NULL);
}

 for(int i=0; i<n; i++){
pthread_join(thrId[i], NULL);
 }
return;
}
```



```
mahan@mahan-N515-52:~/Documents/Week1Changed$ gcc -o Task6 Task6.c -lpthread
mahan@mahan-N515-52:~/Documents/Week1Changed$ ./Task6 2
Thread ID 140399980640000: i=0
Thread ID 140399972247296: i=0
Thread ID 140399980640000: i=1
Thread ID 140399972247296: i=1
Thread ID 140399980640000: i=2
Thread ID 140399972247296: i=2
Thread ID 140399980640000: i=3
Thread ID 140399972247296: i=3
Thread ID 140399980640000: i=4
Thread ID 140399972247296: i=4
Thread ID 140399980640000: i=5
Thread ID 140399972247296: i=5
Thread ID 140399980640000: i=6
Thread ID 140399972247296: i=6
Thread ID 140399980640000: i=7
Thread ID 140399972247296: i=7
Thread ID 140399980640000: i=8
Thread ID 140399972247296: i=8
Thread ID 140399980640000: i=9
Thread ID 140399972247296: i=9
mahan@mahan-N515-52:~/Documents/Week1Changed$
```