INDEX

Sl. No.	Exercise title	Date of performance	Date of correction	Page number	Signature (Internal Supervisor)
1	Perform program for swapping of two numbers without using a temporary variable	9/9/2020		1	
2	Perform program for swapping of two numbers using a temporary variable	9/9/2020		2	
3	Calculate sum of digits of a 5 digit number	9/9/2020		3	
4	Reverse a 5 digit number	10/9/2020		5	
5	Print a new number by adding one to each of a five digit number	10/9/2020		7	
6	Find absolute value of number	10/9/2020		9	
7	Determine whether an year entered is a leap year or not using logical operators	10/9/2020		11	
8	Determine whether an year entered is a leap year or not without using logical operators	10/9/2020		12	
9	Determine whether the character entered is a capital case letter, smaller case letter, a digit or a special symbol using conditional operators	10/9/2020		14	
10	Determine whether the character entered is a capital case letter, smaller case letter, a digit or a special symbol using conditional operators without using conditional operators	10/9/2020		16	
11	Check whether a triangle is valid or not	14/9/2020		18	
12	Find factorial of a number	14/9/2020		19	
13	Generate Fibonacci series from 0 to 500	14/9/2020		20	
14	Prime number generation from 0 to 500	14/9/2020		22	

15	Find the value of one number raised to another number			24	
16	Generate all Armstrong number between 1 and 500	14/9/2020		25	
17	Generate Floyd's triangle	15/9/2020		27	
18	Make menu driven programs with a minimum of 4 options	15/9/2020		29	
19	Write a program using function to calculate the factorial of any integer	15/9/2020		33	
20	Write a function that receives 5 integers and return sum, average and standard deviation of those numbers entered	16/9/2020		35	
21	Use recursive function to obtain the first 25 numbers of a Fibonacci series	16/9/2020		37	
22	To find binary equivalent of a number	16/9/2020		39	
23	To find a number from an array and display how many times it is present	21/9/2020		40	
24	Implement selection sort	21/9/2020		42	
25	Implement Bubble sort	21/9/2020		44	
26	Find the largest number from a 5 x 5 matrix	21/9/2020		46	
27	Perform a 3 x 3 matrix addition	21/9/2020		50	
28	Perform a 3 x 3 matrix multiplication	21/9/2020		53	
29	Find the transpose of a 3x3 matrix	21/9/2020		55	
30	Write a program to extract part of the given string from the specified position	22/9/2020		58	
31	Write a program to sort a set of names stored in an array in alphabetical order	22/9/2020		60	
32	Write a program to delete all vowels from a sentence. Assume that the sentence is not more than 80 characters long.	22/9/2020		62	
33	Create a storage structure for a	22/9/2020		64	

	school library and access the elements				
34	**Read a file and display its contents along with line number before each line**	**23/9/2020**		**72**	
35	**Program to append the contents of one file at the end of another file**	**28/9/2020**		**74**	
36	**Program to copy the content of one file to another**	**28/9/2020**		**76**	
37	**Implement the concept of Arrays using Data structures**	**29/9/2020**		**79**	
38	**Implement the concept of Strings using Data structures**	**29/9/2020**		**82**	
39	**Implement the concept of Stack using Data structures**	**29/9/2020**		**86**	
40	**Implement the concept of Queue using Data structures**	**29/9/2020**		**89**	

Sl. No.	Exercise title	Date of performance	Page number	Signature (Internal Supervisor)
1	Numerical Representation	22/9/2020	92	
2	Beauty of Numbers	22/9/2020	93	
3	More on Numbers	22/9/2020	94	
4	Factorials	22/9/2020	95	
5	String Operations	22/9/2020	96	
6	Recursion	22/9/2020	97	
7	Advanced Arithmatic	22/9/2020	98	
8	Searching and Sorting	22/9/2020	99	
9	Permutation	22/9/2020	100	
10	Sequences	22/9/2020	101	

Final remarks about the Journal:

Date:9/9/2020

Exercise 1: Perform program for swapping of two numbers without using a temporary variable.

Algorithm:

Step 1: Start
Step 2: Read x and y
Step 3: x = x * y;
Step 4: y = x / y;
Step 5: x = x / y;
Step 6: Print x and y
Step 7: Stop

C code:

Untitled2swapping.c | Untitled2temporary.c

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

int main()
 {
   int x, y;                          //read integers x and y
   printf("Enter the value of x and y: ");
   scanf("%d %d", &x, &y);                  //asking the value for x and y
   printf("Before swapping x=%d, y=%d ", x, y);

    x = x * y;
    y = x / y;
    x = x / y;
   printf("After swapping x=%d, y=%d", x, y);
}
```

Output:

```
C:\Users\Prasad\Documents\Untitled2swapping.exe
Enter the value of x and y: 34 54
Before swapping x=34, y=54 After swapping x=54, y=34
--------------------------------
Process exited after 3.213 seconds with return value 25
Press any key to continue . . .
```


Roll No:BID1900

Date:9/9/2020

Exercise 2: Perform program for swapping of two numbers using a temporary variable.

Algorithm:

Step 1 : Start
Step 2 : Declare integer x,y,temp
Step 3 : READ x, y
Step 4 : temp = x
Step 5 :x= y
Step 6 :y = temp
Step 7 : PRINT x, y
Step 8 : Stop

C code:

Untitled2swapping.c [*] Untitled2temporary.c

```c
#include<stdio.h>
#include<conio.h>
int main()
 {
    //using temp as a temporary variable
   int x, y,temp;                          //read intergers x y and temp
   printf("Enter the value of x and y: ");          //ask the value for x and y
   scanf("%d %d", &x, &y);
   printf("Before swapping x=%d, y=%d ", x, y);
   //code begins for program of swapping of two numbers using a temporary variable.
    temp=x;
    x=y;
    y=temp;
   printf("After swapping x=%d, b=%d", x, y);
}
```

Output:

```
C:\Users\Prasad\Documents\Untitled2temporary.exe
Enter the value of x and y: 67
98
Before swapping x=67, y=98 After swapping x=98, b=67
--------------------------------
Process exited after 3.511 seconds with return value 25
Press any key to continue . . .
```


Roll No:BID1900

Date:9/9/2020

Exercise 3:Calculate sum of digits of a 5 digit number

Algorithm:

Step 1 : Start
Step 2 : Declare integer num, num1,num2,num3,num4,num5, sum
Step 3 : READ num
Step 4 : num1 = num % 10
Step 5 : num = num/10
Step 6 : num2 = num%10
Step 7 : num = num/10
Step 8 : num3 = num % 10
Step 9 : num = num/10
Step 10 : num4 = num%10
Step 11 : num = num/10
Step 12 : num5 = num%10
Step 13 : num = num/10
Step 14 : sum= num1 + num2+ num3+ num4+ num5
Step 15 : PRINT sum
Step 16 : Stop

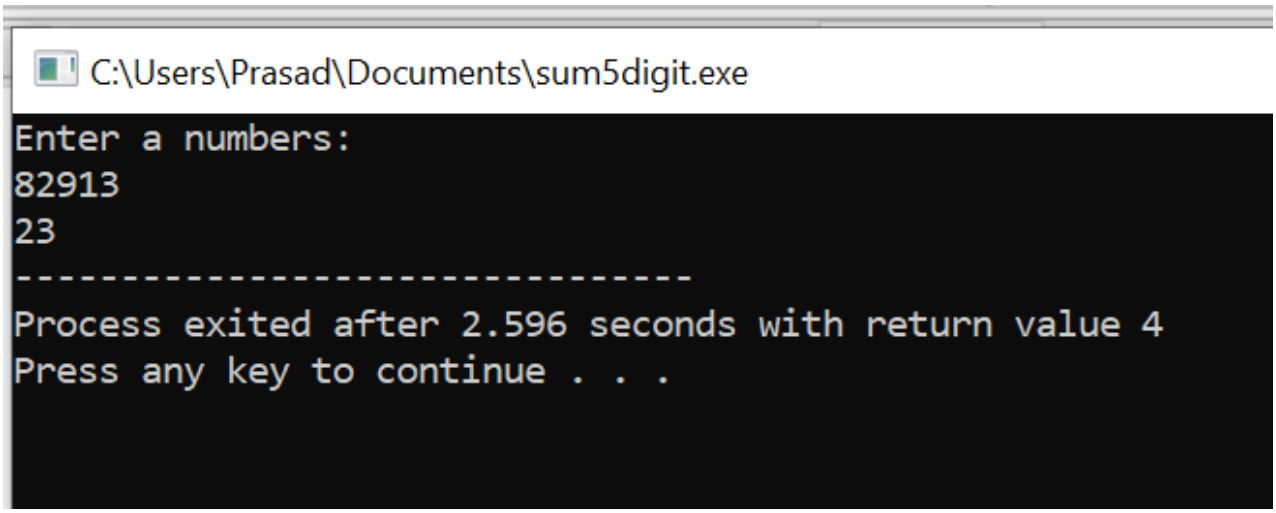
C code:

Untitled2swapping.c | [*] Untitled2temporary.c | sum5digit.c

```c
#include<stdio.h>
#include<conio.h>
void main()
{
// Declare integer num,num1,num2,num3,num4,num5, sum
      int num,num1,num2,num3,num4,num5, sum ;
      printf("Enter a numbers:\n");
       scanf("%d", &num);                                  // READ num
      num1 = num % 10;
      num = num/10;
      num2 = num%10;
      num = num/10;
      num3 = num % 10;
      num = num/10;
      num4 = num%10;
      num = num/10;
     num5 = num%10;
     num = num/10;
     sum= num1+num2+num3+num4+num5;
     printf("%d   " ,sum);

}
```


Roll No:BID1900

Output:



Roll No:BID1900

Date:10/9/2020

Exercise 4: Reverse a 5 digit number.

Algorithm:

Step 1 : Start
Step 2 : Declare long int num,num1,num2,num3,num4,num5, rev
Step 3 : READ num
Step 4 : Num1 = num%10
Step 5 : Num2=(num/10)%10
Step 6 : Num3=(num/100)%10
Step 7 : Num4=(num/1000)%10
Step 8 : Num5=(num/10000)%10
Step 9 : Rev=(num1*10000+num2*1000+num3*100+num4*10+num5)
Step 10 : PRINT rev
Step 11 : Stop

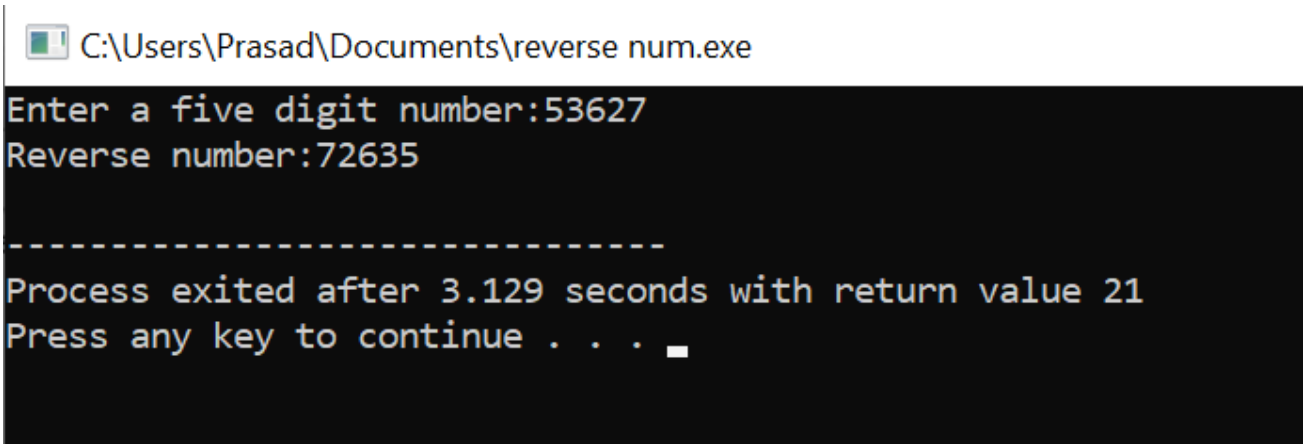
C code:

Untitled2swapping.c | [*] Untitled2temporary.c | sum5digit.c | reverse num.c

```c
#include<stdio.h>
#include<conio.h>
void main()
{
    //Declare long int num,num1,num2,num3,num4,num5, rev
    long int num,num1,num2,num3,num4,num5,rev;
    printf("Enter the five digit number:\n");
    scanf("%ld",&num);              //READ num
    num1 = num%10;
    num2=(num/10)%10;
    num3=(num/100)%10;
    num4=(num/1000)%10;
    num5=(num/10000)%10;
    rev=(num1*10000+num2*1000+num3*100+num4*10+num5);
    printf("Reverse=%ld", rev);
}
```


Roll No:BID1900

Output:



Roll No:BID1900

Date:10/9/2020

Exercise 5: Print a new number by adding one to each of a five digit number.

Algorithm:

Step 1 : Start
Step 2 : Declare long int num,num1,num2,num3,num4,num5, newnum
Step 3 : READ num
Step 4 : Num1 = (num%10) + 1
Step 5 : Num2=((num/10)%10)+1
Step 6 : Num3=((num/100)%10)+1
Step 7 : Num4=((num/1000)%10)+1
Step 8 : Num5=((num/10000)%10)+1
Step 9 :newnum=(num5*10000+num4*1000+num3*100+num2*10+num1)
Step 10 : PRINT newnum
Step 11 : Stop

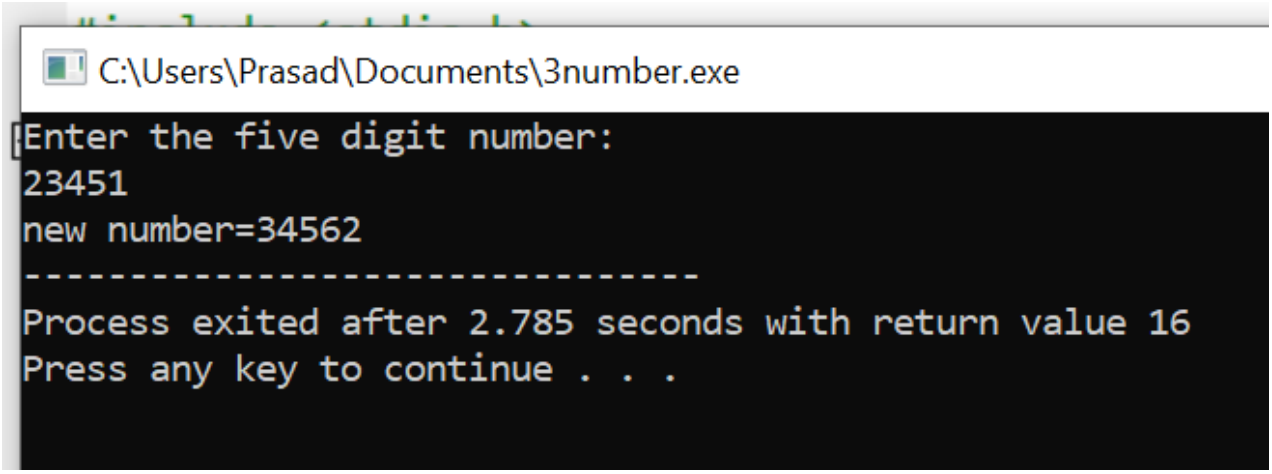
C code:

Untitled2swapping.c | [*] Untitled2temporary.c | sum5digit.c | reverse num.c | [*] 3number.c

```c
#include<conio.h>
#include <stdio.h>
void main()
{
    //Declare long int num,num1,num2,num3,num4,num5, newnum
    long int num,num1,num2,num3,num4,num5,newnum;
    printf("Enter the five digit number:\n");
    scanf("%ld",&num);                    //READ num
    num1 =(num%10)+1;
    num2=((num/10)%10)+1;
    num3=((num/100)%10)+1;
    num4=((num/1000)%10)+1;
    num5=((num/10000)%10)+1;
    newnum=(num5*10000+num4*1000+num3*100+num2*10+num1);
    printf("new number=%ld", newnum);        //PRINT newnum
}

```


Roll No:BID1900

Output:



Roll No:BID1900

Date:10/9/2020

Exercise 6: Find absolute value of a number

Algorithm:

Step 1 : Start
Step 2 : Declare int num
Step 3 : READ num
Step 4 : If (num<0) , then
Step 5 : PRINT num
Step 6 :num = num*-1;
Step 7 : PRINT num
Step 8 : Stop

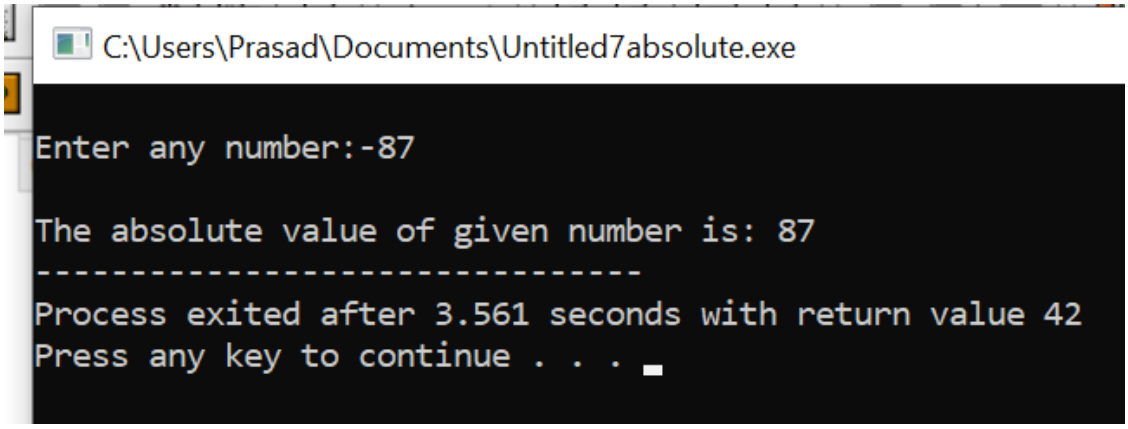
C code:

Untitled2swapping.c | [*] Untitled2temporary.c | sum5digit.c | reverse num.c | [*] 3number.c | Untitled7absolute.c

```c
#include<stdio.h>
#include<conio.h>
void main()
{
  int num;                  //Declare int num , READ num


  printf("\nEnter any number:");

  scanf("%d",&num);      //reading entered value from user

  if(num< 0)
    num = num*-1;
    printf("\nThe absolute value of given number is: %d", num);

}
```


Roll No:BID1900

Output:



Roll No:BID1900

Date:10/9/2020

Exercise 7: Determine whether an year entered is a leap year or not using logical operators.

Algorithm:

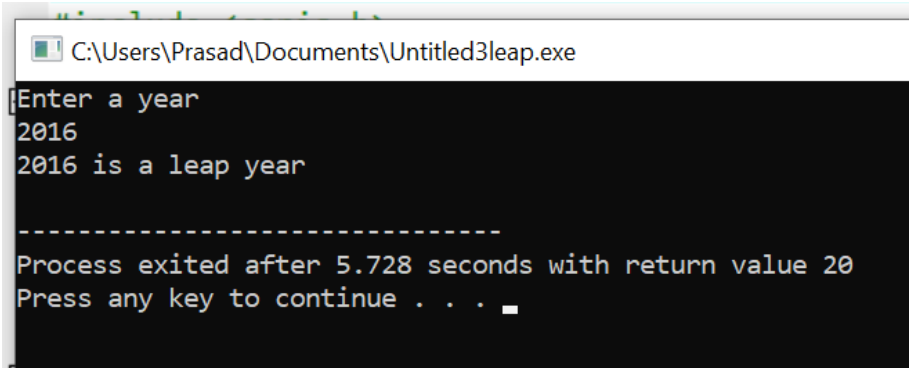
Step 1 : Start
Step 2 : Declare int year
Step 3 : READ year
Step 4 : if(year%100 == 0 && year%400 == 0) || (year%100 != 0 && year%4 == 0))
Step 5 : PRINT "is a leap year"
Step 6 : else PRINT " is not a leap year"
Step 7 : Stop

C code:

Untitled2swapping.c | [*] Untitled2temporary.c | sum5digit.c | reverse num.c | [*] 3number.c | Untitled7absolute.c | [*] Untitled3leap.c

```c
#include <stdio.h>
#include <conio.h>
int main()
{
    int year;             //Declare int year

    printf("Enter a year\n");
    scanf("%d", &year);          //READ year

    if( (year%100 == 0 && year%400 == 0) ||  (year%100 != 0 && year%4   == 0)   )
    {
        printf("%d is a leap year\n", year);
    }
    else
    {
        printf("%d is not a leap year\n", year);
    }
```

Output



Roll No:BID1900

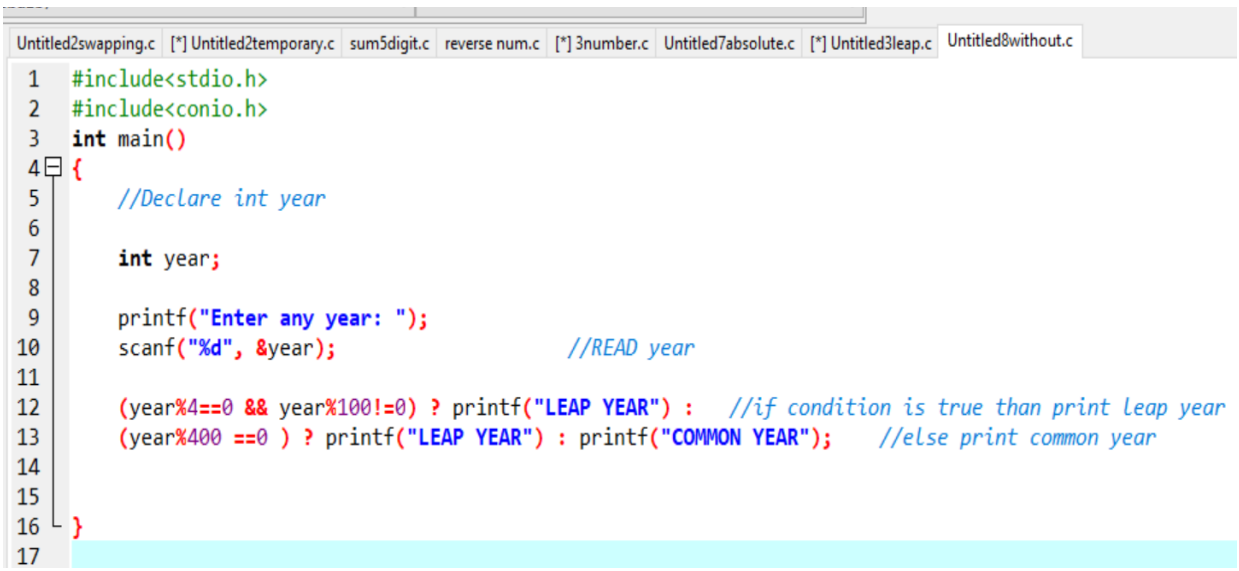
Date:10/9/2020

Exercise 8: Determine whether an year entered is a leap year or not without using logical operators.

Algorithm:

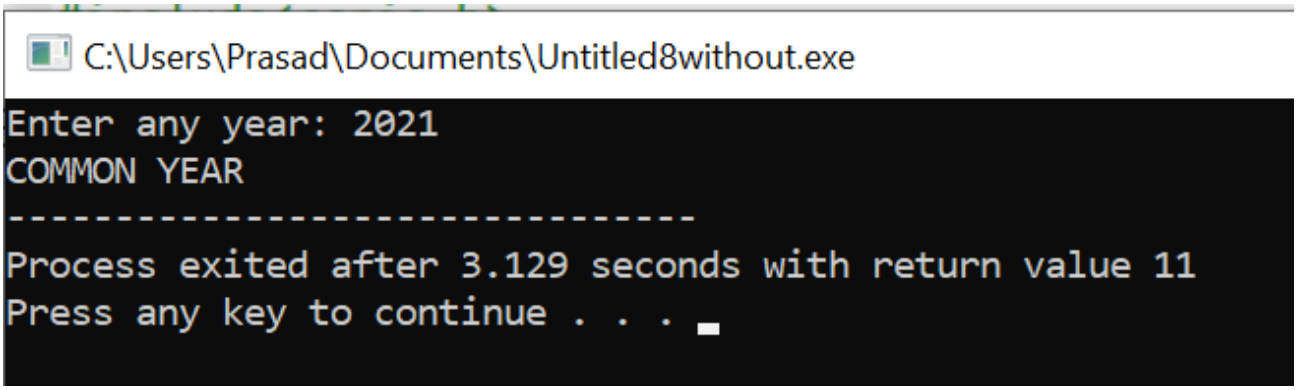
Algorithm:
Step 1 :Start
Step 2 :Declare int year
Step 3 :READ year
Step 4 :if (year%4==0 && year%100!=0) ? printf("LEAP YEAR") :(year%400 ==0) ?
Step 5 :PRINT “leap year”
Step 6 :else PRINT “common year”
Step 7 :Stop

C code:



Roll No:BID1900

Output:



Roll No:BID1900

Date:10/9/2020

Exercise 9: Determine whether the character entered is a capital case letter, smaller case letter, a digit or a special symbolusing conditional operators

Algorithm:

Step 1 :Start
Step 2 :Declare char ‘ch’
Step 3 :Read ch
Step 4: (ch>=65 &&ch<=90), then Print capital letter.
Step 5 : else if(ch>=97 &&ch<=122), then Print Small letter
Step 6 : else if(ch>=48 &&ch<=57), then Print Digit
Step 7 : else if((ch>=0 &&ch<=47) || (ch>=58&&ch<=64) || (ch>=91 &&ch<=96) || (ch>=123 &&ch<=127)), then Print special symbol
Step 8 : Stop

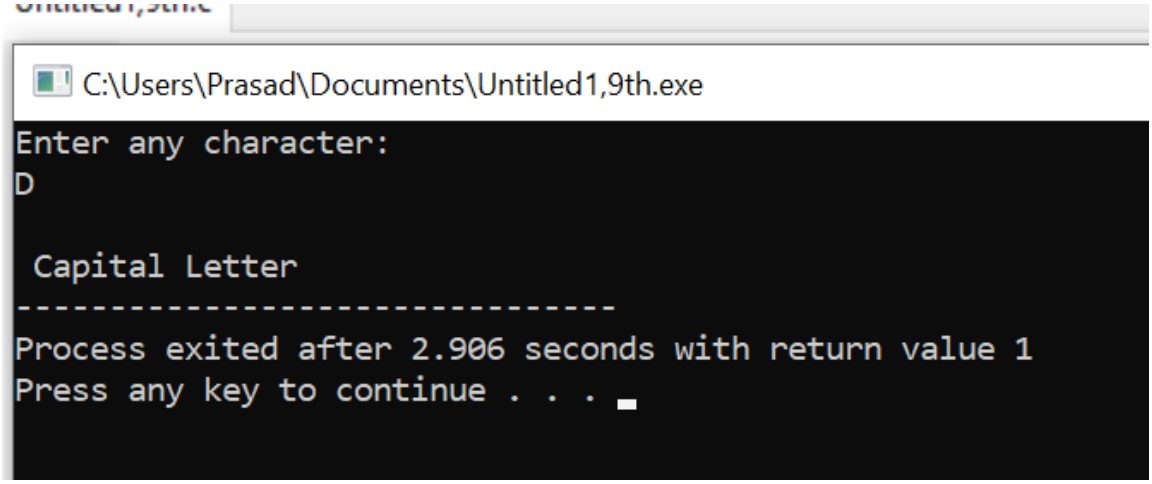
C code:

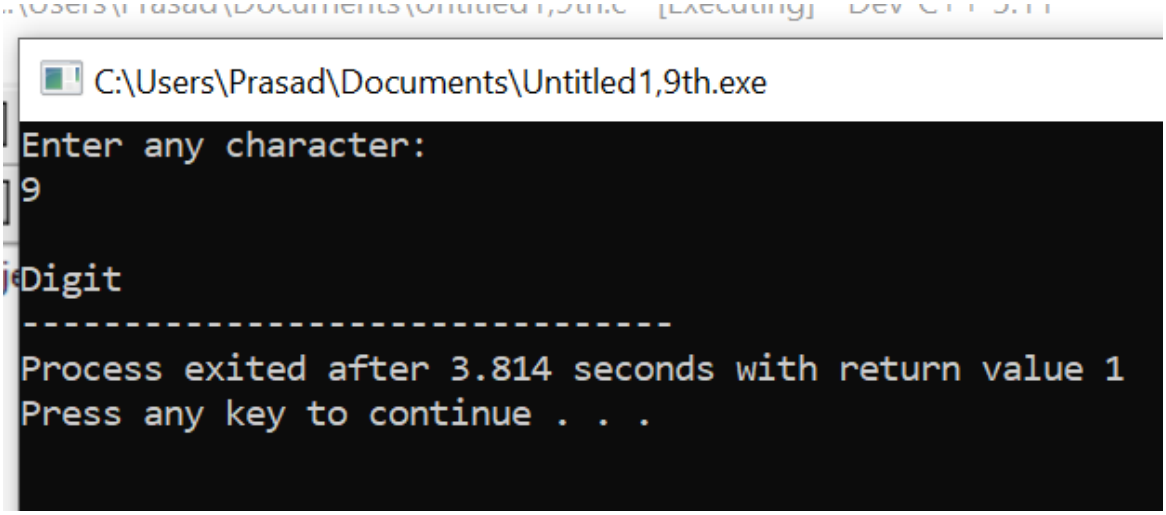
Untitled2swapping.c | [*] Untitled2temporary.c | sum5digit.c | reverse num.c | [*] 3number.c | Untitled7absolute.c | [*] Untitled3leap.c | Untitled8without.c | [*] Untitled1,9th.c

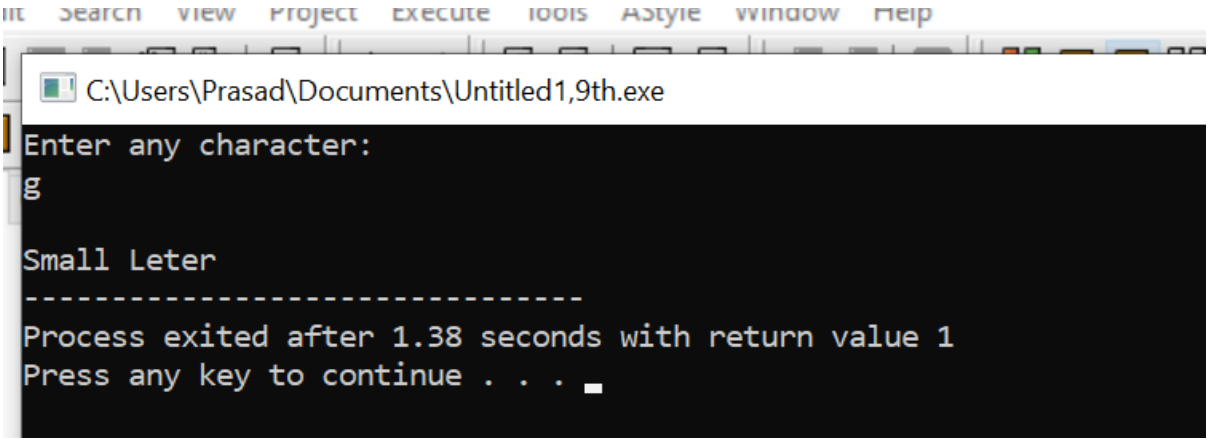
```c
#include <stdio.h>
#include<conio.h>

void main()
{
char ch;                //Declare char 'ch'
printf("Enter any character:\n");
scanf("%c",&ch);         // Read ch
(ch>=65 && ch<=90?printf("\n Capital Letter"):printf(" "));         //ch>=65 && ch<=90 print cappital letter
(ch>=97&&ch<=122?printf("\nSmall Leter"):printf(" "));               //ch>=97&&ch<=122 print small letter
(ch>=48&&ch<=57?printf("\nDigit"):printf(" ") );                    //ch>=48&&ch<=57 print digit
((ch>=0&&ch<=47)||(ch>=58&&ch<=64)||(ch>=91&&ch<=96)||(ch>=123&&ch<127)?printf("Special Symbol"):printf(" "));

}
```


Roll No:BID1900

Output:







Roll No:BID1900

Date:10/9/2020

Exercise 10 :Determine whether the character entered is a capital case letter, smaller case letter, a digit or a special symbolusing conditional operators without using conditional operators

Algorithm:

Step 1 :Start

Step 2 :Declare char ‘ch’

Step 3 :Read ch

Step 4: Conditions are applied and checked, if(ch>=65 &&ch<=90), then Print capital letter.

Step 5 : else if(ch>=97 &&ch<=122), then Print Small letter

Step 6 : else if(ch>=48 &&ch<=57), then Print Digit

Step 7 : else if((ch>=0 &&ch<=47) || (ch>=58&&ch<=64) || (ch>=91 &&ch<=96) || (ch>=123 &&ch<=127)), then Print special symbol

Step 8 : Stop

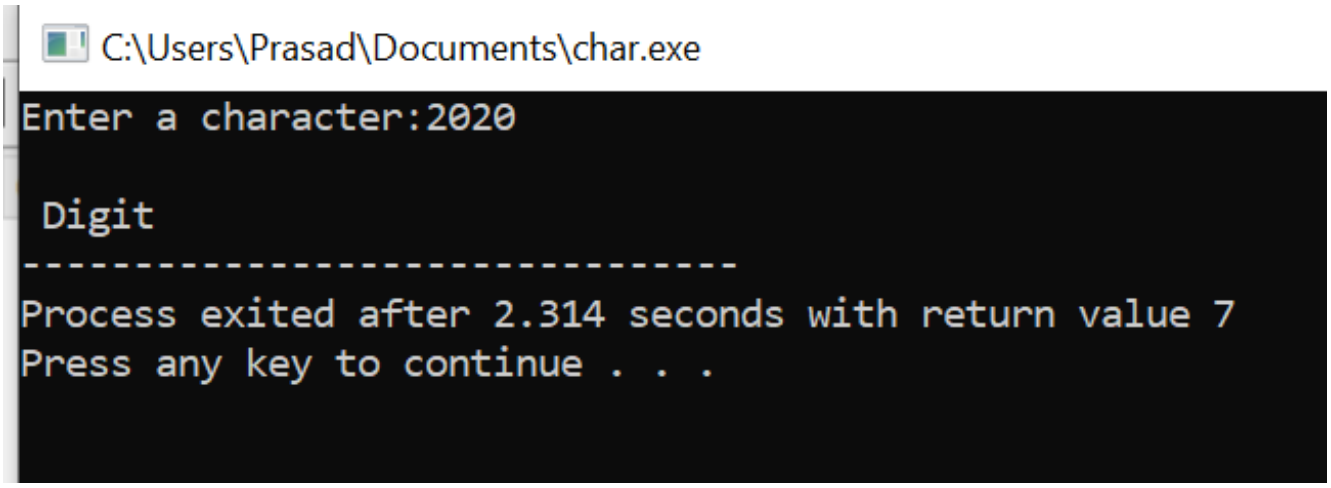
C code:

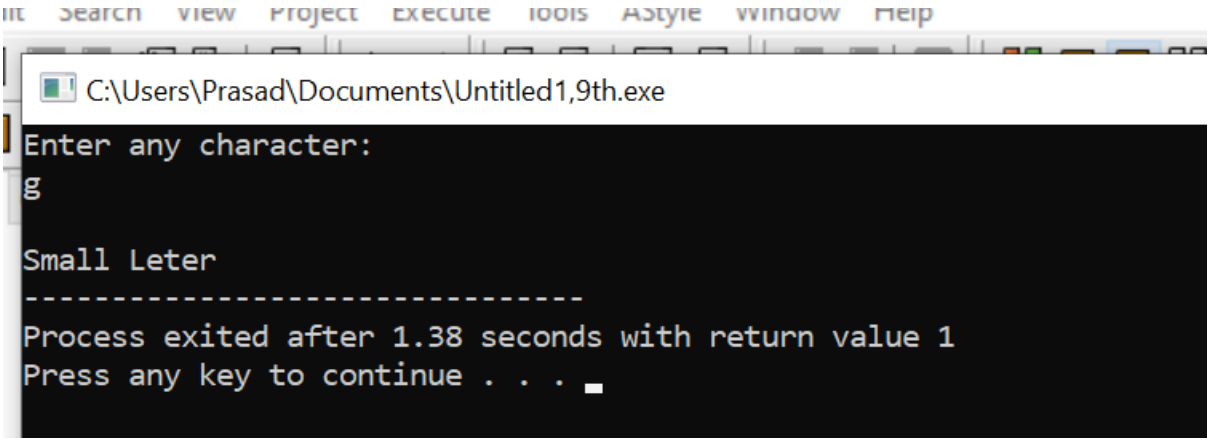
Untitled2swapping.c | [*] Untitled2temporary.c | sum5digit.c | reverse num.c | [*] 3number.c | Untitled7absolute.c | [*] Untitled3leap

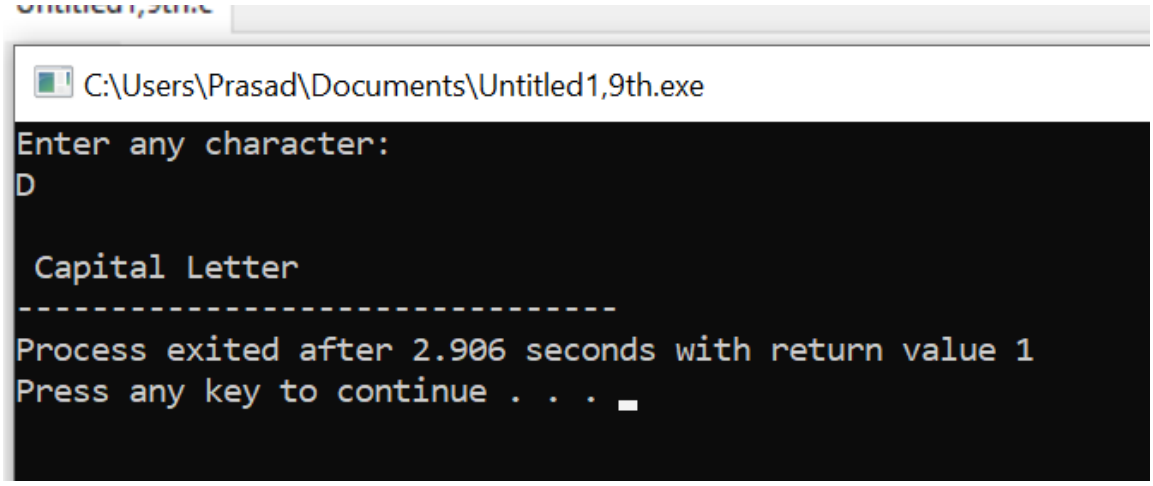
```c
#include<stdio.h>
#include<conio.h>
void main()
{
char ch;                    //Declare char 'ch'
printf("Enter a character:\n");
scanf("%c",&ch);        //Read ch
if(ch>=65 && ch<=90)                //Conditions are applied and checked,
{
printf("\n Capital letter");   //if(ch>=65 && ch<=90), then Print capital letter.
}
else if(ch>=97 && ch<=122)
{
printf("\n Small letter"); //else if(ch>=97 && ch<=122), then Print Small letter
}
else if(ch>=48 && ch<=57)
{
printf("\n Digit");  //else if(ch>=48 && ch<=57), then Print Digit
}
else if((ch>=0 && ch<=47) || (ch>=58&& ch<=64) || (ch>=91 && ch<=96) || (ch>=123 && ch<=127))
{
printf("\n Special symbol");
}
```


Roll No:BID1900

Output:







Roll No:BID1900

Date:14/9/2020

Exercise 11: Check whether a triangle is valid or not

Algorithm:

Step 1 :Start

Step 2 :Declare int g1, g2, g3;

Step 3 :Read int g1, g2, g3;

Step 4 : Conditions are applied and checked, if(g1 + g2 > g3) && (g2 + g3 > g1) && (g1 + g3 > g2))

then PRINT “This is a Valid Traingle”

Step 5 : Else, then PRINT ”This is an Invalid Triangle"

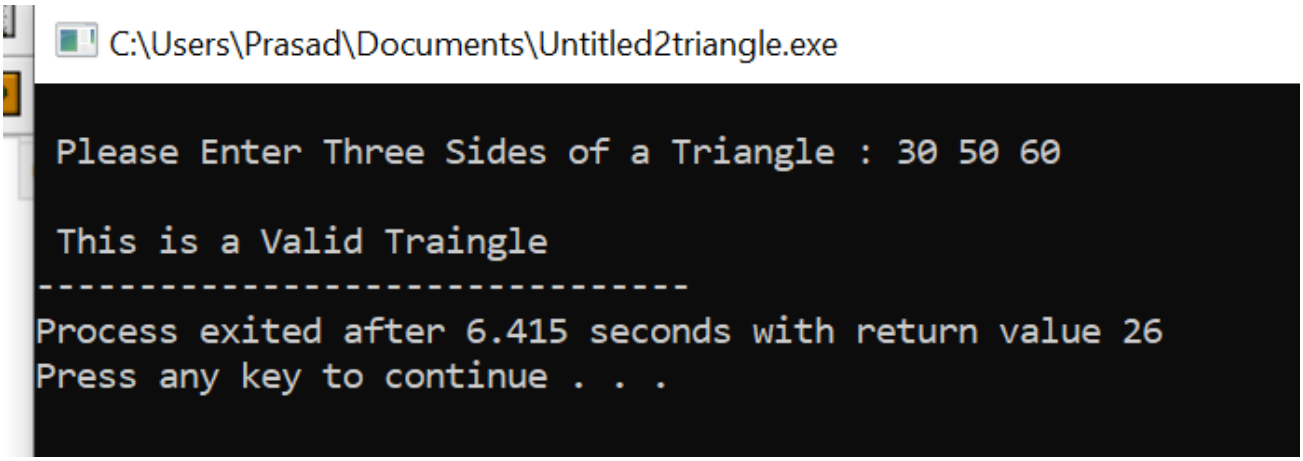
Step 6 : Stop

C code:

Untitled2swapping.c | [*] Untitled2temporary.c | sum5digit.c | reverse num.c | [*] 3number.c | Untitled7absolute.c | [*] Untitled3leap.c | Untitled8without.c | [*] Unti

```c
#include<stdio.h>
#include<conio.h>
int main()
{
    int g1, g2, g3;    //Declare int g1, g2, g3;
    printf("\n Please Enter Three Sides of a Triangle : ");
    scanf("%d%d%d", &g1, &g2, &g3);            // Read int g1, g2, g3;

    if( (g1 + g2 > g3) && (g2 + g3 > g1) && (g1 + g3 > g2) )     //Conditions are applied and checked,
    {
        printf("\n This is a Valid Traingle");  //if (g1 + g2 > g3) && (g2 + g3 > g1) && (g1 + g3 > g2)
    }                                           //then PRINT "This is a Valid Traingle"
    else
    {
        printf("\n This is an Invalid Triangle");      //Else, then PRINT "This is an Invalid Triangle"
    }

}
```

Output:



Roll No:BID1900

Date:14/9/2020

Exercise 12: Find factorial of a number.

Algorithm:

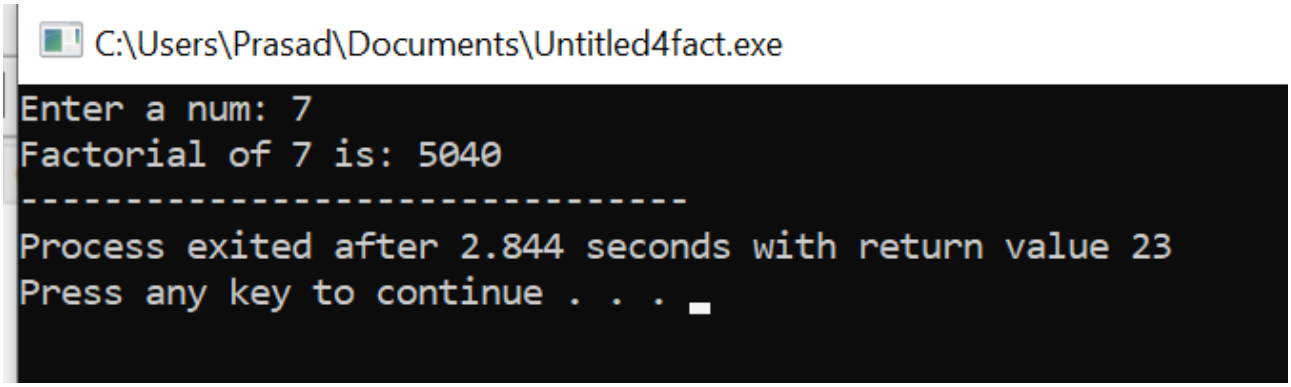
Step 1 :Start
Step 2 :Declare i,g=1,num;
Step 3 :Read num
Step 4 :Use for loop for finding factorial
Step 5 : PRINT Factorial
Step 6 : Stop

C code:

Untitled1 [*] Untitled4fact.c

```c
#include<stdio.h>
#include<conio.h>
int main()
{
 int i,g=1,num;    //Declare i,g=1,num;
 printf("Enter a num: ");      //Read num
  scanf("%d",&num);
    for(i=1;i<=num;i++)    //Use for loop for finding factorial
    {
    g=g*i;
  }
  printf("Factorial of %d is: %d",num,g);
}

```

Output:



Roll No:BID1900

Date:14/9/2020

Exercise 13: Generate Fibonacci series from 0 to 500.

Algorithm:

Step 1: START
Step 2: Take integer variable g1, g2, nextnum
Step 3: Set g1 = 0, g2 = 1
Step 4: DISPLAY g1
Step 5 :nextnum= g1 + g2
Step 6: Set g1= g2, g2= nextnum
Step 7 : repeat for 500 times ;
Step 8: STOP.

C code:

Untitled1 | [*] Untitled4fact.c | [*] Untitled5fabo.c

```c
#include <stdio.h>
#include <conio.h>
int main()
{
    int i, g1 = 0, g2 = 1, nextnum;        //Take integer variable g1, g2, nextnum
    printf("Fibonacci Series: ");       //Set g1 = 0, g2 = 1

    for (i = 1; i <= 500; ++i)
    {
        printf("%d, ", g1);     //DISPLAY g1
        nextnum = g1 + g2;
        g1 = g2;
        g2 = nextnum;
    }

}
```


Roll No:BID1900

Output:

Roll No:BID1900

Date:14/9/2020

Exercise 14: Prime number generation from 0 to 500.

Algorithm:

1. Start
2. initialize low=0, high =500,I,flag
3. 'the numbers between 0 and 500 ' is displayed .
4. Until low <= high the loop will run .
5. If not goto step 13
7. if low <= 1
8. increment low
9. for(i=2;i<=low/2;++i)
10. divide low with I if the remainder is 0 then the value flag becomes 1.
11. if the value of flag is 0 then print low.
12. increment low
13. Stop

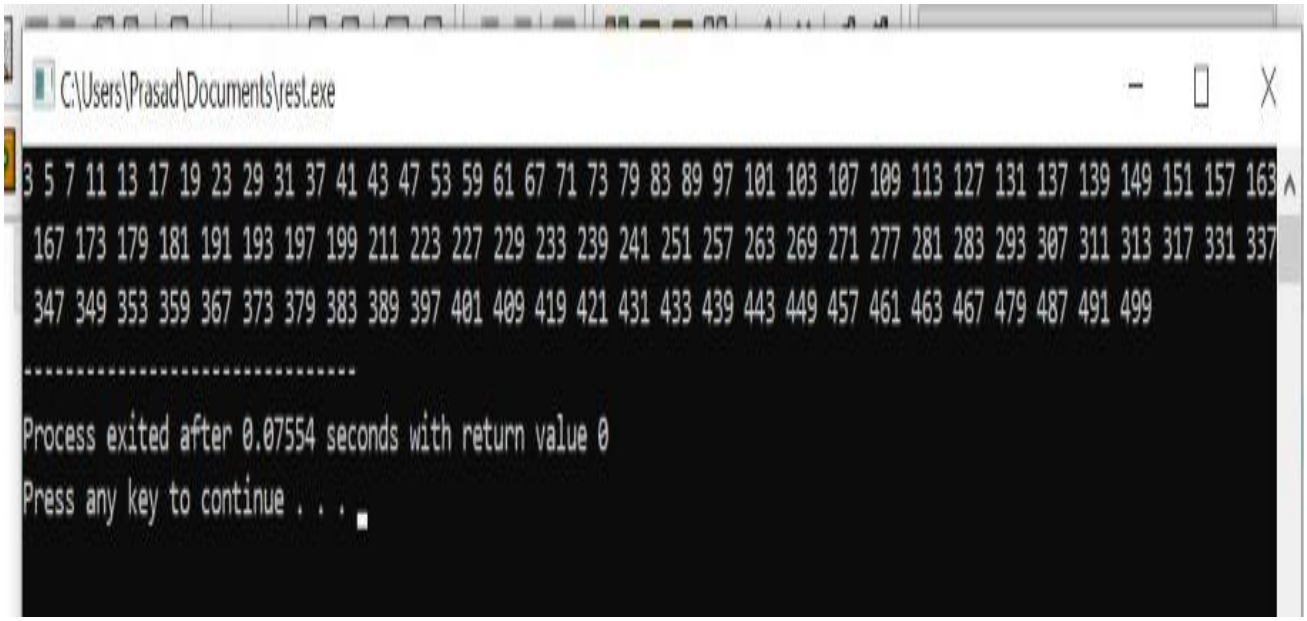
C code:

rest.c

```
#include<stdio.h>
#include<conio.h>
#include <math.h>
void main()
{
int i,j;
for(i=3;i<=500;i++)
{
for(j=2;j<i;j++)
{
if(i%j==0)
break;
if(j>sqrt(i))
{
printf("%d ",i);
break;
}
}
```

Output:

Roll No:BID1900



Roll No:BID1900

Date:14/9/2020

Exercise 15: Find the value of one number raised to another number.

Algorithm:

Step 1 :Start
Step 2 : Declare integers g1, g2, i, num=1;
Step 3 : READ g1 and g2
Step 4 : Use loop , for(i=1;i<=g2;i++)
Step 5 :num=num*g1;
Step 6 : PRINT num
Step 7 : Stop

Code:

Untitled1 | [*] Untitled4fact.c | [*] Untitled5fabo.c | [*] Untitled3prime.c | Untitled6raised.c

```c
#include<stdio.h>
#include<conio.h>
int main()
{
    int g1, g2, i, num=1;      //Declare integers g1, g2, i, num=1;
        printf("Enter any two numbers ");
    scanf("%d%d",&g1,&g2);          // READ g1 and g2
                                 // Use loop , for(i=1;i<=g2;i++)
    for(i=1;i<=g2;i++)
    {
        num=num*g1;
    }
    printf("%d raised to %d is %d",g1,g2,num);

}
```

Output:

```
C:\Users\Prasad\Documents\Untitled6raised.exe
Enter any two numbers 2 3
2 raised to 3 is 8
--------------------------------
Process exited after 5.185 seconds with return value 18
Press any key to continue . . .
```


Roll No:BID1900

Date:14/9/2020

Exercise 16: Generate all Armstrong number between 1 and 500

Algorithm:

Step 1 :Start
Step 2 : Declare variables num,r,sum,temp
Step 3 : Use nested loop , for(num=1;num<=500;num++)
Step 4 : while(temp!=0)
Step 5 : use condition if(sum==num)
Step 6 : PRINT numbers
Step 7 : Stop

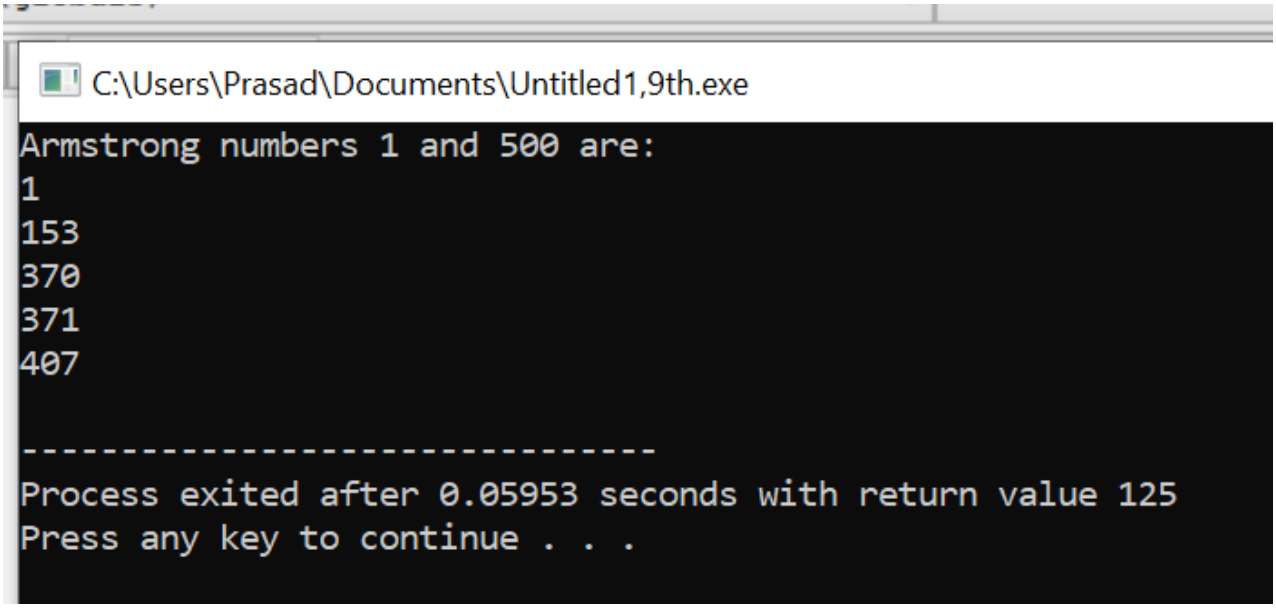
C code:

Untitled1 | [*] Untitled4fact.c | [*] Untitled5fabo.c | [*] Untitled3prime.c | Untitled6raised.c | [*] Untitled1,9th.c

```c
#include<conio.h>
#include<stdio.h>
void main()
{
int num,r,sum,temp;          //Declare variables num,r,sum,temp
printf("Armstrong numbers 1 and 500 are:\n");
for(num=1;num<=500;num++)          //Use nested loop , for(num=1;num<=500;num++)
{
temp=num;
sum=0;
while(temp!=0)
{
r=temp%10;                  //while(temp!=0)
temp=temp/10;
sum=sum+(r*r*r);
}
if(sum==num)          //use condition if(sum==num)
{
printf("%d\n",num);
}
}
}
```


Roll No:BID1900

Output:



Roll No:BID1900

Date:15/9/2020

Exercise 17: Generate Floyd's triangle

Algorithm:

Step 1 :Start
Step 2: Declare int n, i, k, g = 1;
Step 3 :Use nested loop , for(i=1;i<=n;i++)
Step 4 :Make inner iteration
Step 5 :Print g
Step 6 :Increment g
Step 7 :Stop

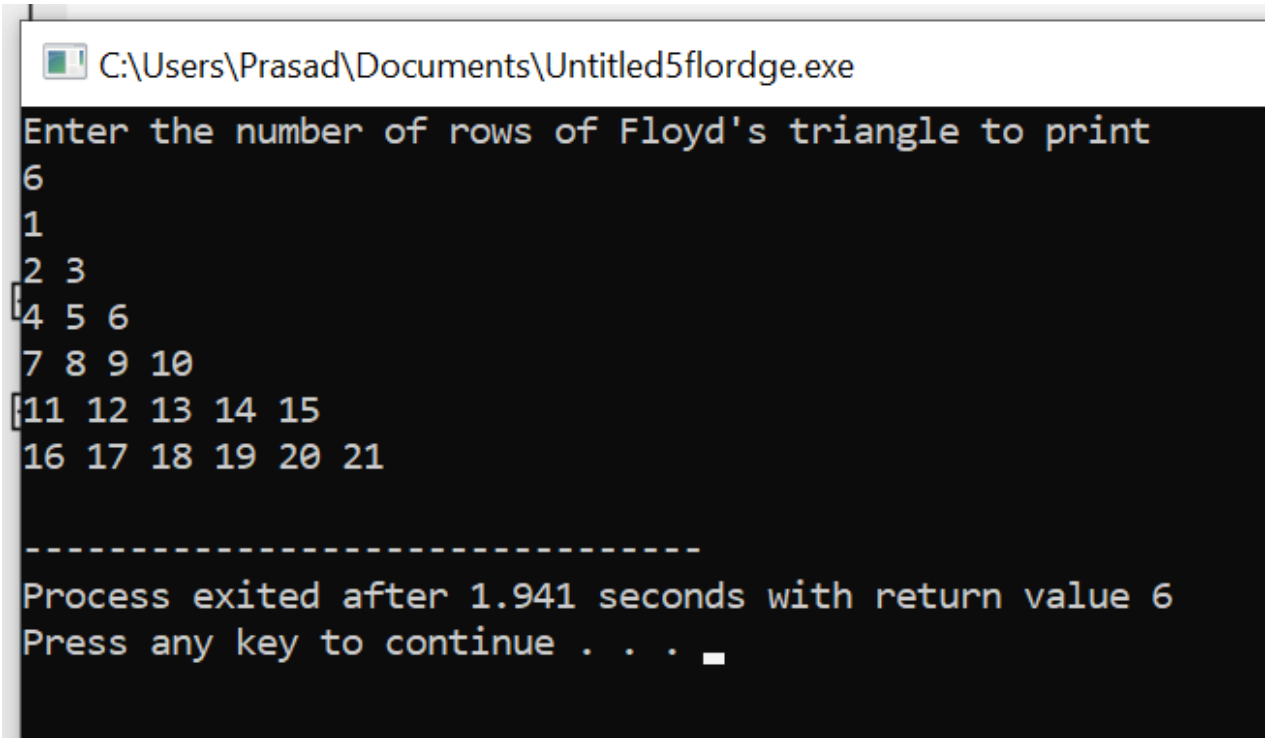
C code:

Untitled1 | [*] Untitled4fact.c | [*] Untitled5fabo.c | [*] Untitled3prime.c | Untitled6raised.c | Untitled1,9th.c | [*] Untit

```c
#include <stdio.h>
#include<conio.h>
int main()
{
    int n, i,  k, g = 1;              //Declare int n, i,  k, g = 1;
     printf("Enter the number of rows of Floyd's triangle to print\n");
    scanf("%d", &n);
    //Use nested loop , for(i=1;i<=n;i++)
    for (i = 1; i <= n; i++)
    {
        for (k = 1; k <= i; k++)          //Make inner iteration
        {
            printf("%d ",g);              //Print g
            g++;                          //increment g
        }
        printf("\n");
    }
    }
```


Roll No:BID1900

Output:



Roll No:BID1900

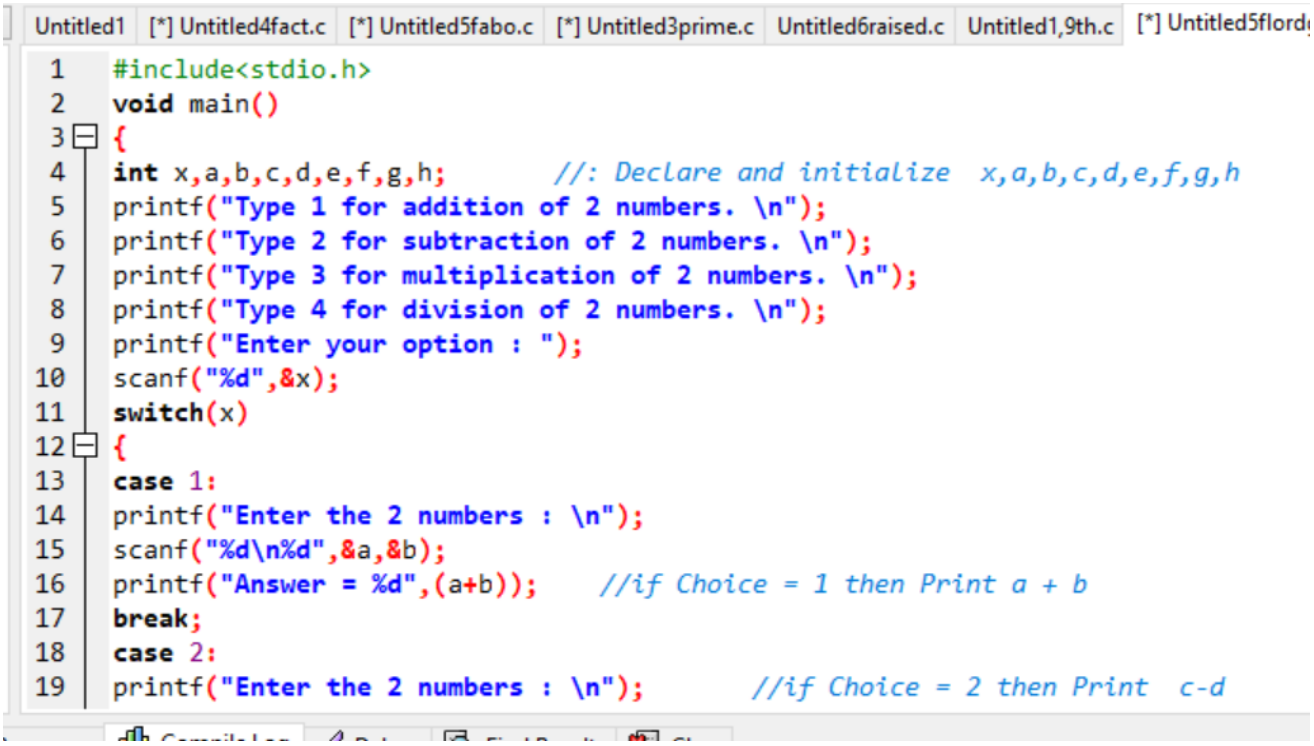
Date:15/9/2020

Exercise 18: Make menu driven programs with a minimum of 4 options

Algorithm:

Step 1 :Start
Step 2 :Declare and initialize x,a,b,c,d,e,f,g,h
Step 3 : if Choice = 1 then Print a + b
Step 4 : if Choice = 2 then Print c-d
Step 5 : if Choice = 3 then Print e * f
Step 6 : if Choice = 4 then Print g / h
Step 7 :Stop

C code:



Roll No:BID1900

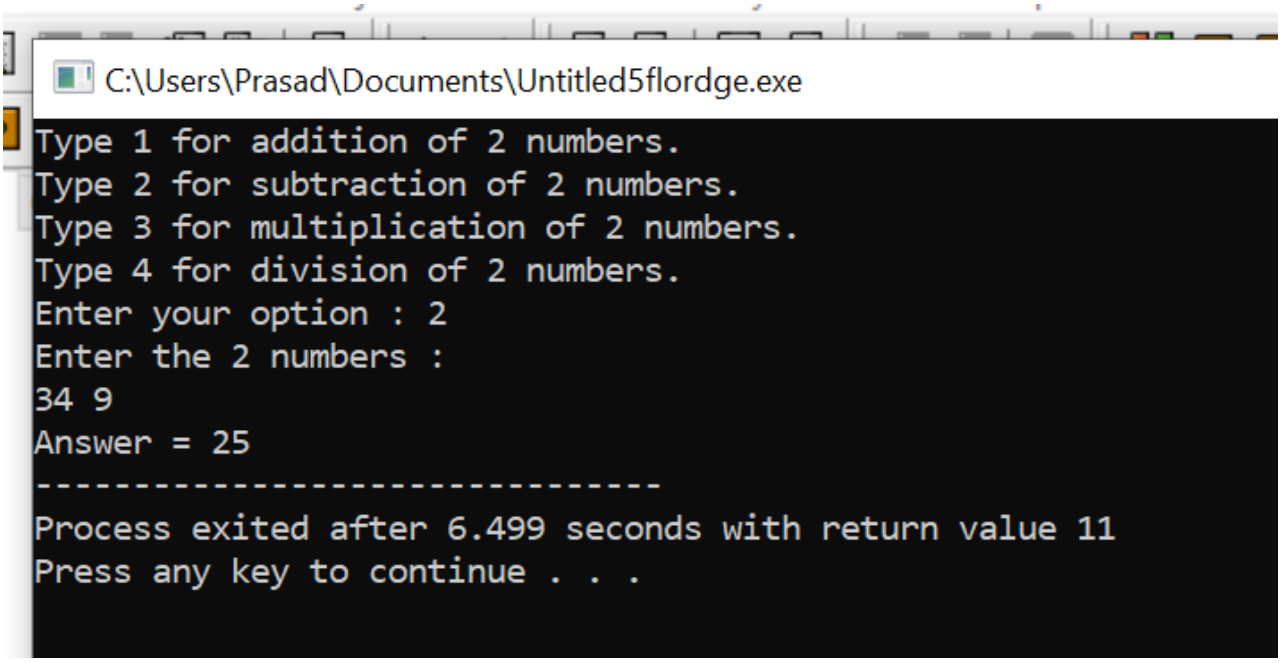
Untitled1 | [*] Untitled4fact.c | [*] Untitled5fabo.c | [*] Untitled3prime.c | Untitled6raised.c | Untitled1,9th.c | [*] Untit

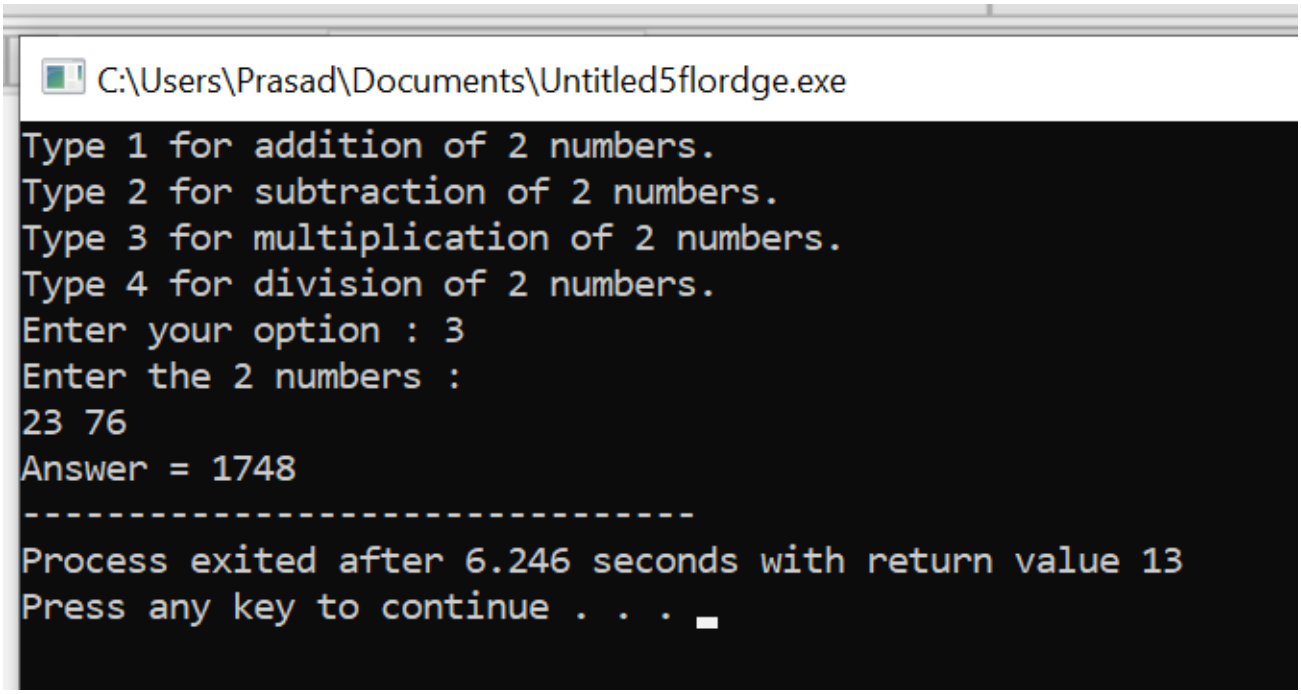
```
18  case 2:
19  printf("Enter the 2 numbers : \n");        //if Choice = 2 then Print  c-
20  scanf("%d\n%d",&c,&d);
21  printf("Answer = %d",(c-d));
22  break;
23  case 3:
24  printf("Enter the 2 numbers : \n");     //if Choice = 3 then Print  e * f
25  scanf("%d\n%d",&e,&f);
26  printf("Answer = %d",(e*f));
27  break;
28  case 4:
29  printf("Enter the 2 numbers : \n");
30  scanf("%d\n%d",&g,&h);
31  printf("Answer = %d",(g/h));           //if Choice = 4 then Print  g / h
32  break;
33  default:
34  printf("Invalid.");
35  }
36  }
```

Output:

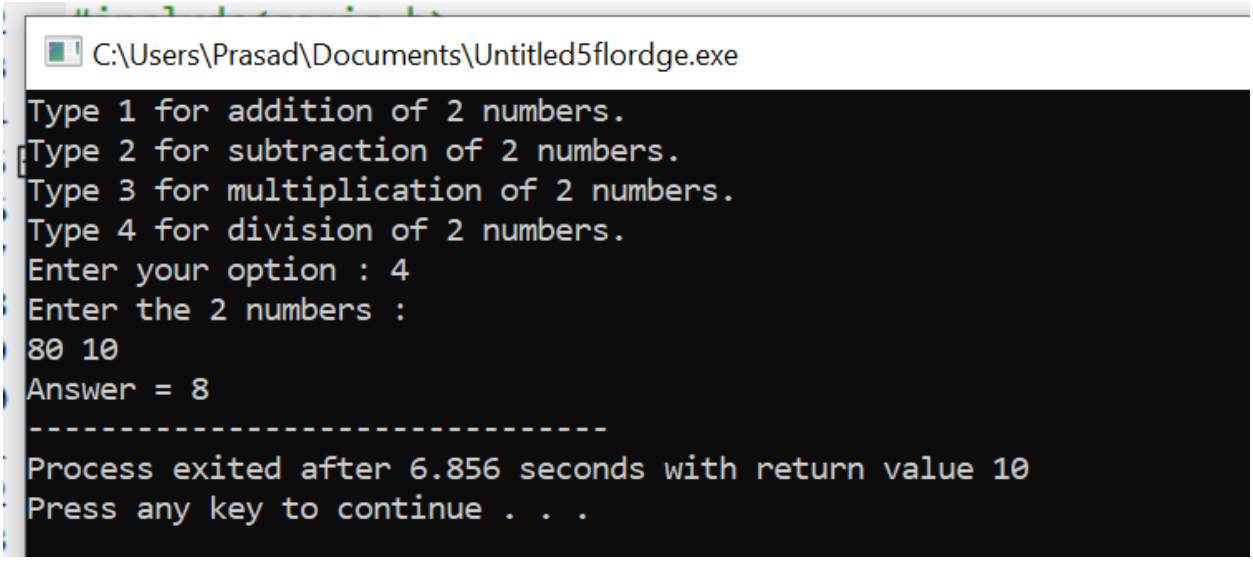
```
C:\Users\Prasad\Documents\Untitled5flordge.exe
Type 1 for addition of 2 numbers.
Type 2 for subtraction of 2 numbers.
Type 3 for multiplication of 2 numbers.
Type 4 for division of 2 numbers.
Enter your option : 1
Enter the 2 numbers :
43 5
Answer = 48
--------------------------------
Process exited after 9.952 seconds with return value 11
Press any key to continue . . .
```


Roll No:BID1900





Roll No:BID1900



Roll No:BID1900

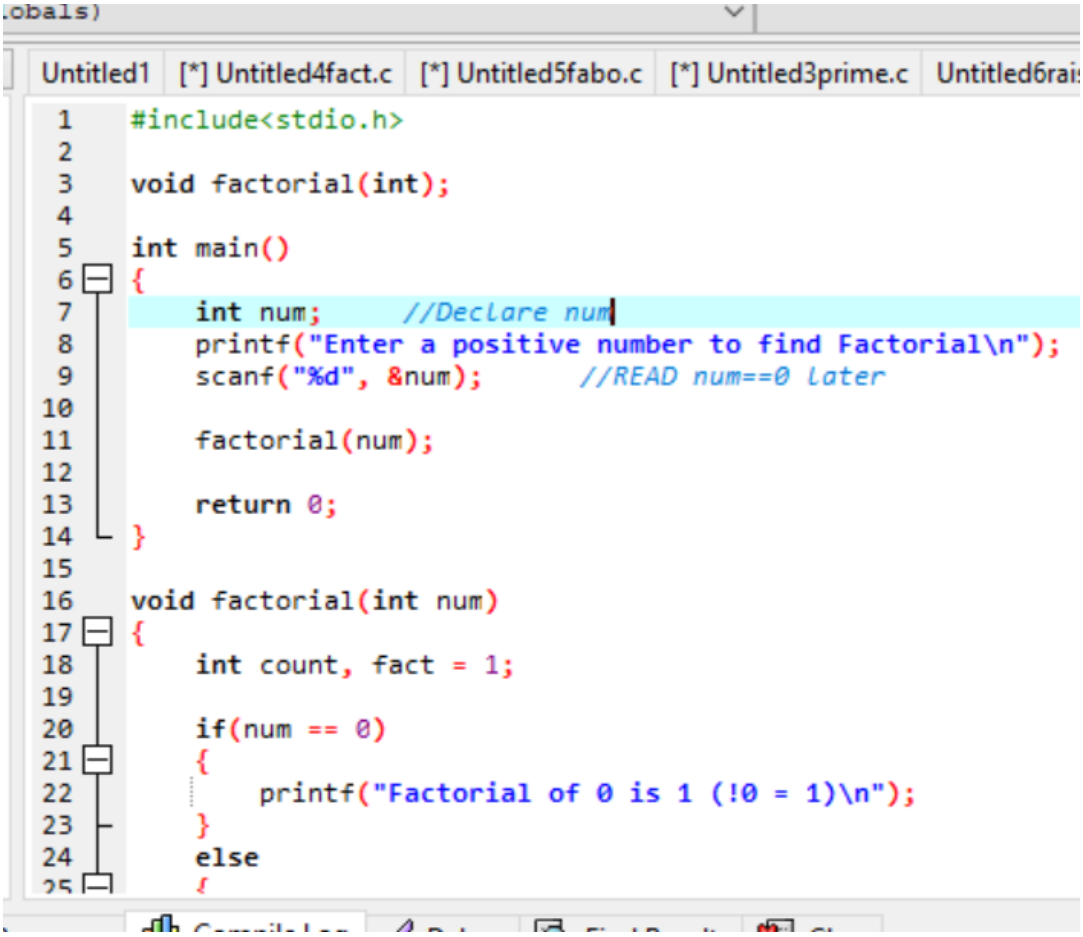
Date:15/9/2020

Exercise 19: Write a program using function to calculate the factorial any Integer

Algorithm:

Step 1 :Start
Step 2 :Declare num
Step 3 :READ num==0
Step 4 :Declare fact
Step 5 :use for loop (count = 1; count <= num; count++)
Step 6: fact = fact * count;
Step 7 :Stop

C code:

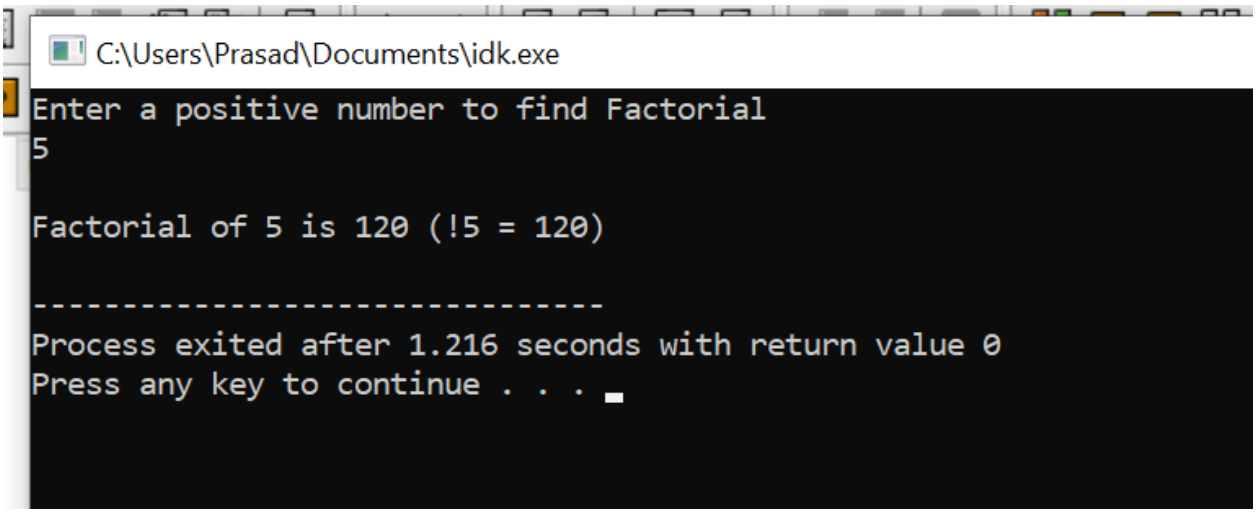


Roll No:BID1900

Untitled1 | [*] Untitled4fact.c | [*] Untitled5fabo.c | [*] Untitled3prime.c | Untitled6raised.c | Untitled1,9th.c | [*] Untitled5flordge.c | idk.c

```c

    factorial(num);

    return 0;
}

void factorial(int num)
{
    int count, fact = 1;

    if(num == 0)
    {
        printf("Factorial of 0 is 1 (!0 = 1)\n");
    }
    else
    {
        for(count = 1; count <= num; count++)    //use for loop (count = 1; count <= num; count++)
        {
            fact = fact * count;      //Declare fact
        }

        printf("\nFactorial of %d is %d (!%d = %d)\n", num, fact, num, fact);
    }
}
```

Output:



Roll No:BID1900

Date:16/9/2020

Exercise 20 :Write a function that receives 5 integers and return sum, average and standard deviation of those numbers entered

Algorithm:

Step 1 :Start
Step 2 : Declaring function *sum, float *av, float *std
Step 3 :Declare variables in main function
Step 4 : Taking Input
Step 5 : Passing the values
Step 6 : Defining the Function after the main function
Step 7 : PRINT the results
Step 8 :Stop

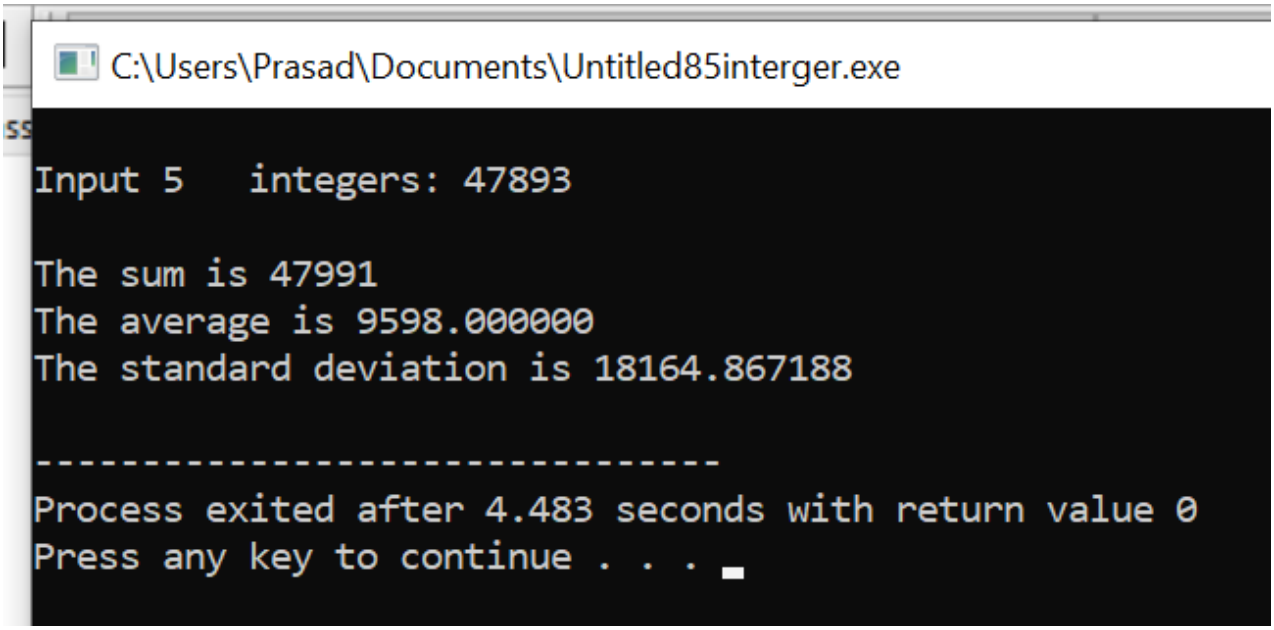
C code:

Untitled1 | [*] Untitled4fact.c | [*] Untitled5fabo.c | [*] Untitled3prime.c | Untitled6raised.c | Untitled1,9th.c | [*] Untitled5flordge.c

```c
#include <stdio.h>
#include <math.h>
#include<conio.h>
//Declaring function *sum, float *av, float *std
void func(int a, int b, int c, int d, int e, int *sum, float *av, float *std)
{
 *sum = a + b + c + d + e;
 *av = *sum / 5;
 *std = sqrt((((a - *av)*(a - *av)) + ((b - *av)*(b - *av)) + ((c - *av)*(c - *av))) / 5.0);
}
//Declare variables in main function
int  main( )
{
 float ave, std;
 int a, b, c, d, e, sum;
 printf("\nInput 5   integers: ");       //Taking Input
 scanf("%d,%d,%d,%d,%d", &a, &b, &c, &d, &e);     //Passing the value
 func(a, b, c, d, e, &sum, &ave, &std);         // Defining the Function after the main function
 printf("\nThe sum is %d\nThe average is %f\nThe standard deviation is %f\n", sum, ave, std);
 return 0;
}
```


Roll No:BID1900

Output:



Roll No:BID1900

Date:16/9/2020

Exercise 21: Use recursive function to obtain the first 25 numbers of a Fibonacci series

Algorithm:

Step 1 :Start
Step 2 : Declaring function fibo
Step 3 :Declare variables in main function
Step 4 : Taking Input
Step 5 : Passing the values
Step 6 : Defining the Function after the main function
Step 7 : By using Loop which runs 25 times
Step 8 : PRINT the values
Step 9 : Again passing values
Step 10 : Returning values to main()
Step 11 : PRINT results
Step 12 :Stop

C code:

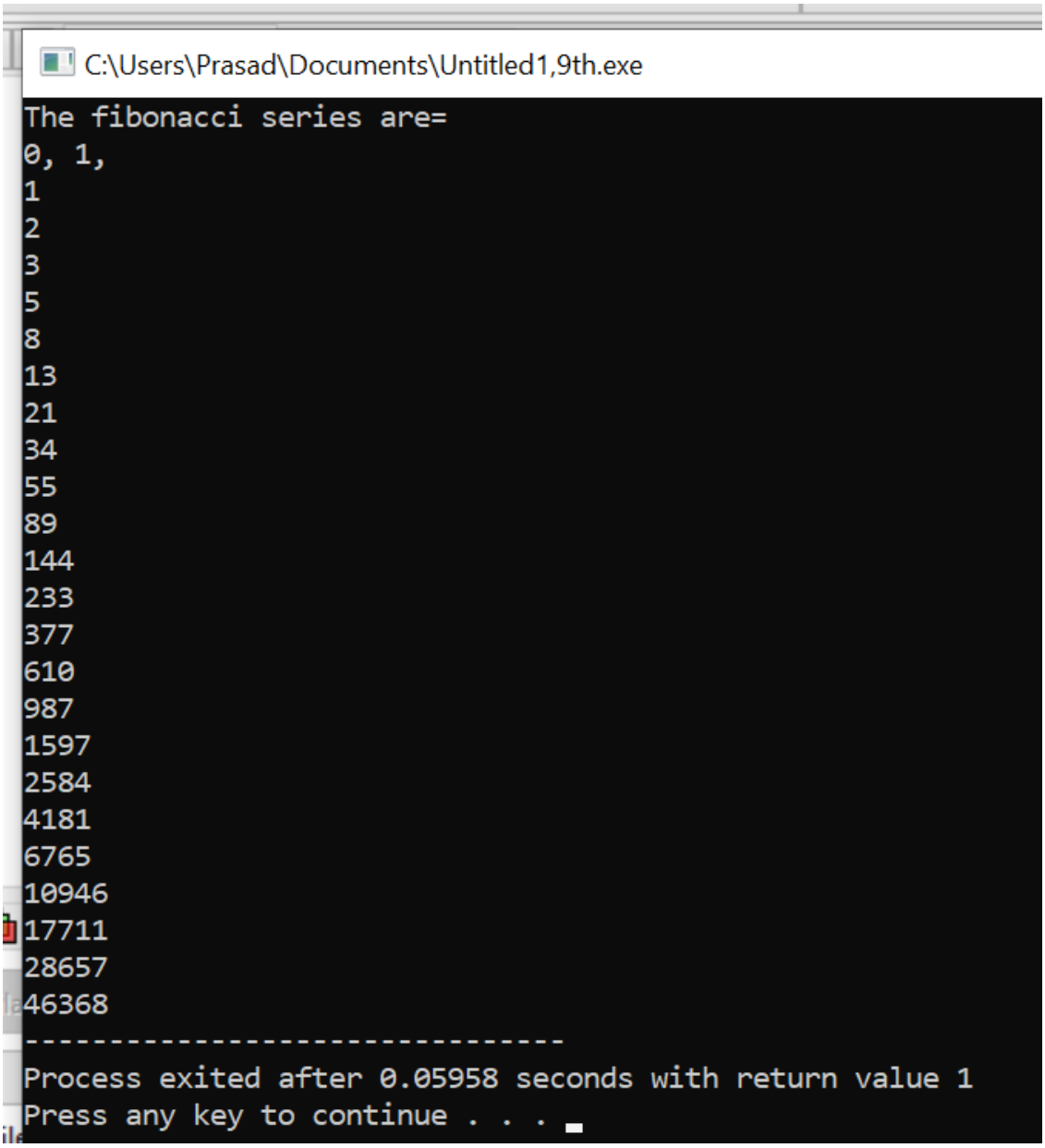
Untitled1 | [*] Untitled4fact.c | [*] Untitled5fabo.c | [*] Untitled3prime.c | Untitled6raised.c

```c
#include<stdio.h>
#include<conio.h>
long int fibo(long int, long int);  //Declaring function fibo
void main()
//Declare variables in main function
{
long int i,t,prev=0,curr=1,next;
printf("The fibonacci series are=\n"); //Taking Input
printf("%ld, %ld, ",prev,curr);
fibo(prev,curr);
//Passing the values
}
long int fibo(long int prev, long int curr)
//Defining the Function after the main function
{
static long int count=2;
long int nre,next;
//By using Loop which runs 25 times
if(count<25)
{
next=prev+curr;
printf("\n%ld, ",next);     //PRINT the values
count=count+1;      //Again passing values
```


Roll No:BID1900

```
22      printf("\n%ld, ",next);     //PRINT the values
23      count=count+1;      //Again passing values
24      fibo(curr,next);
25  }
26  //Returning values to main()
27  else
28  return 1;
29  }
```

Output:



Roll No:BID1900

Date:16/9/2020

Exercise 22:To find binary equivalent of a number

Algorithm:

Step 1 : Start
Step 2 : Declaring the array
Step 3 : Taking Input
Step 4 : By using loops
While(n!=0) : Continue till quotient becomes o
for(j=i-1;j>=0;j--) : Printing the array in reverse order to get the binary equivalent
Step 5 : PRINT the results
Step 6 : Stop

C code:

Untitled1 | [*] Untitled4fact.c | [*] Untitled5fabo.c | [*] Untitled3prime.c | Untitled6raised.c | Untitled1,9th.c | [*] Untitled5flordge.c | idk.c | [*] U

```c
#include <stdio.h>
void main()
{
int n,j;
int i=0;
int a[100];   ////Declaring the array
printf("Enter the number:\n"); //Taking Input
scanf("%d",&n);
while(n!=0)    //By using loops
{                 //Continue till quotient becomes o
a[i]=n%2;
i++;
n=n/2;
}
for(j=i-1;j>=0;j--)     //Printing the array in reverse order to get the binary equivalent
{
printf("%d",a[j]);
}
}
```

Output:

```
C:\Users\Prasad\Documents\Untitled1,9th.exe
Enter the number:
33
100001
--------------------------------
Process exited after 3.525 seconds with return value 1
Press any key to continue . . .
```


Roll No:BID1900

Date:21/9/2020

Exercise 23: To find a number from an array and display how many times it is present

Algorithm:

Step 1 : Start
Step 2 : Declaring the array of size 10
Step 3 : Taking Input of array elements
Step 4 : Take the number which is to be found
Step 5 : By using loop , for(j=0,j<10;j++)
Step 6 : By using conditional operators if(b[j]==a)
Step 7 : PRINT the number
Step 8 : Stop

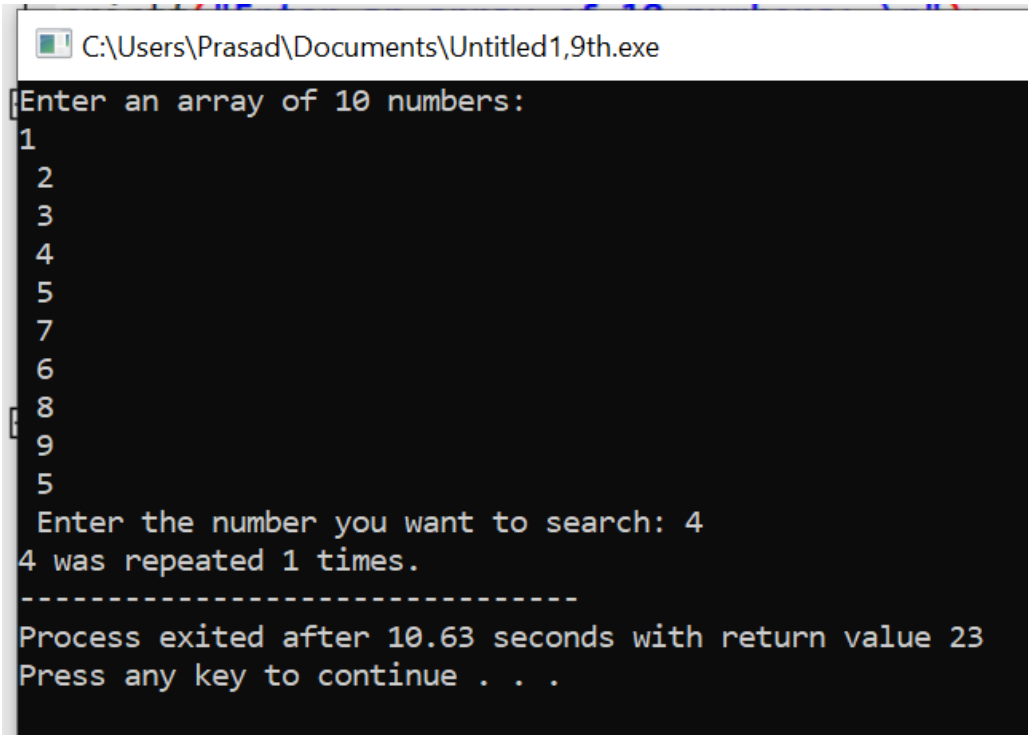
C code:

Untitled1 | [*] Untitled4fact.c | [*] Untitled5fabo.c | [*] Untitled3prime.c | Untitled6raised.c | [*] Untitled1,9th.c | [*] Untitled5flord

```c
#include<stdio.h>
void main()
{
int a,b[10],i,j,c=0; //Declaring the array of size 10
printf("Enter an array of 10 numbers: \n");     //Taking Input of array elements
for(i=0;i<10;i++)
{
scanf("%d",&b[i]);          //Take the number which is to be found
printf(" ");
}
printf("Enter the number you want to search: ");
scanf("%d",&a);
for(j=0;j<10;j++)               //By using loop , for(j=0,j<10;j++)
{
if(b[j]==a)          //By  using conditional operators if(b[j]==a)
c++;
}
printf("%d was repeated %d times.",a,c);          //PRINT the number
}
```


Roll No:BID1900

Output:



Roll No:BID1900

Date:21/9/2020

Exercise 24: Implement selection sort

Algorithm:

Step 1 : Start

Step 2 : Declaring the array of size 5 and other variables

Step 3 : Taking Input of array elements

Step 4 : By using nested loop ,

for(c=0;c<5;c++)

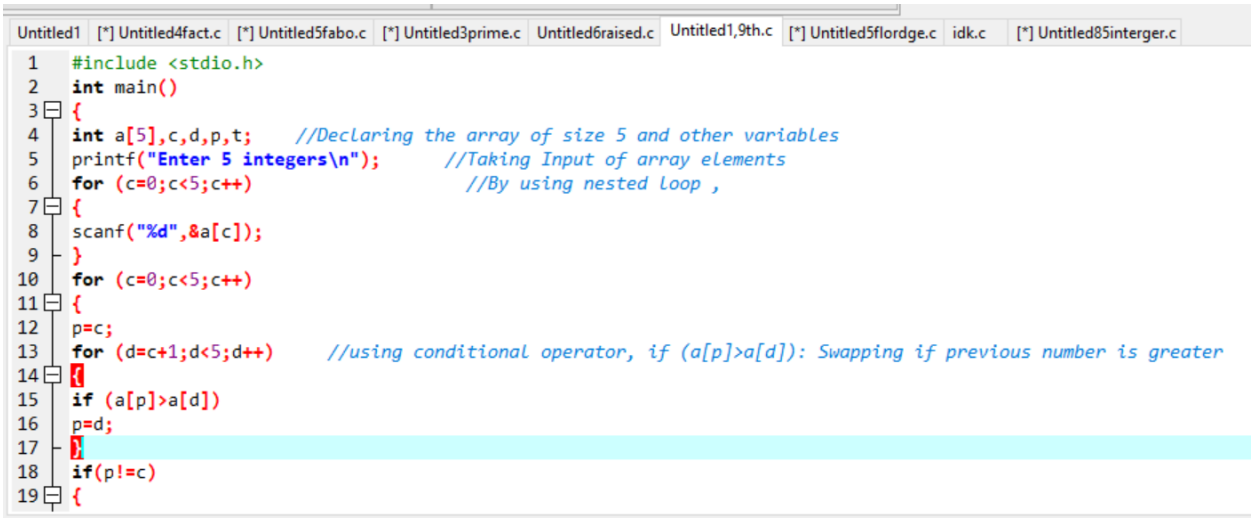
for(d=c+1;d<5;d++)): using conditional operator, if (a[p]>a[d]): Swapping if previous number is greater

Step 5 : Use another loop , for(c=0;c<5;c++): For Printing array in ascending order

Step 6 : PRINT the result

Step 7 : Stop

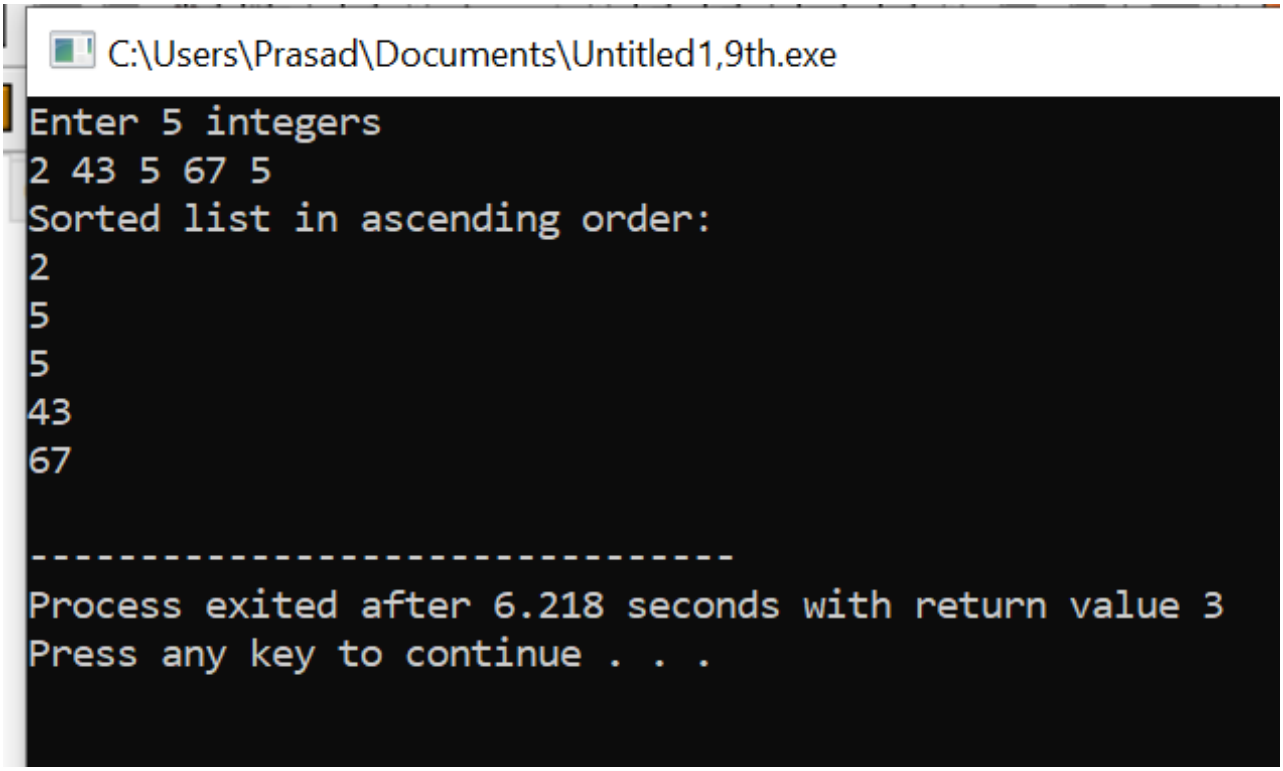
C code:



Roll No:BID1900

```
19  {
20  t=a[c];
21  a[c]=a[p];
22  a[p]=t;
23  }
24  }
25  printf("Sorted list in ascending order:\n");
26  for (c=0;c<5;c++)
27  //Use another loop , for(c=0;c<5;c++): For Printing array in ascending order
28  printf("%d\n",a[c]);      //PRINT the result
29  }
30
```

Output:



Roll No:BID1900

Date:21/9/2020

Exercise 25: Implement Bubble sort

Algorithm:

Step 1 : Start
Step 2 : Declaring the array of size 5 and other variables
Step 3 : Taking Input of array elements
Step 4 : By using nested loop ,
for(c=0;c<4;c++)
for(d=0;d<5-c-1;d++): using conditional operator, if(a[d]>a[d+1]): Swapping numbers if 1'st value is greater than
Step 5 : Use another loof , for(c=0;c<5;c++): For Printing array in ascending order
Step 6 : PRINT the result
Step 7 : Stop

C code:

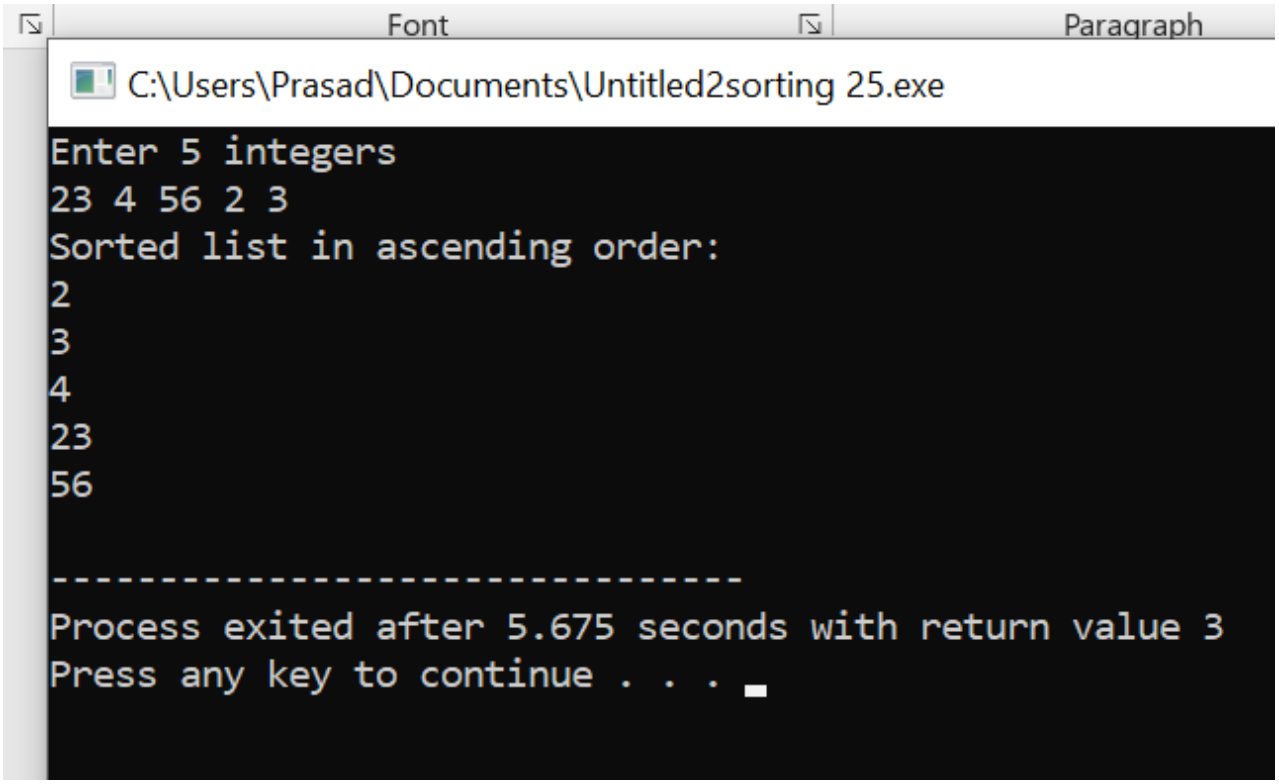
Untitled1 | [*] Untitled4fact.c | [*] Untitled5fabo.c | [*] Untitled3prime.c | Untitled6raised.c | Untitled1,9th.c

```c
#include <stdio.h>
int main()
{
int a[5],c,d,t;
//Declaring the array of size 5 and other variables
printf("Enter 5 integers\n");     //Taking Input of array elements
for (c=0;c<5;c++)          //By using nested loop
scanf("%d",&a[c]);
for (c=0;c<4;c++)       //For Printing array in ascending order
{
    //using conditional operator
for (d=0;d<5-c-1;d++)
{
//if(a[d]>a[d+1]): Swapping numbers if 1'st value is greater than
if(a[d]>a[d+1])
{
t=a[d];
a[d]=a[d+1];
a[d+1]=t;
```


Roll No:BID1900

```
17      t=a[d];
18      a[d]=a[d+1];
19      a[d+1]=t;
20    }
21    }
22    }
23    printf("Sorted list in ascending order:\n");
24    for (c=0;c<5;c++)
25    printf("%d\n",a[c]);
26    }
```

Output:



Roll No:BID1900

Date:21/9/2020

Exercise 26: Find the largest number from a 5 x 5 matrix.

Algorithm:

Step 1: Start
Step 2: Declaring the array of size and other variables
Step 3: Taking Input of array elements
Step 4: By using nested loop ,Update largest number
Step 5: Use another loof , For Printing array in ascending order
Step 6: PRINT the result
Step 7: Stop

C code:

Untitled1 | [*] Untitled4fact.c | [*] Untitled5fabo.c | [*] Untitled3prime.c | Untitled6raised.c | Untitled1,9th.c | [*] Untitled5flordge.c | idk.c | [*] Untitled8

```c
#include<stdio.h>
#include<conio.h>
void main()
{
   int m, n, c, d, matrix[10][10], maximum;   //Declaring the array of size and other variable
   printf("Enter the number of rows and columns of matrix\n"); //Taking Input of array elements
   scanf("%d%d",&m,&n);
   printf("Enter the elements of matrix\n");
   //By using nested loop , Update largest number
   for( c = 0 ; c < m ; c++ )
   {
      for( d = 0 ; d < n ; d++ )
      {
         scanf("%d",&matrix[c][d]);
      }
   }
      printf("\nMatrix is :\n");
    for(c=0;c< m;c++)
    {
```

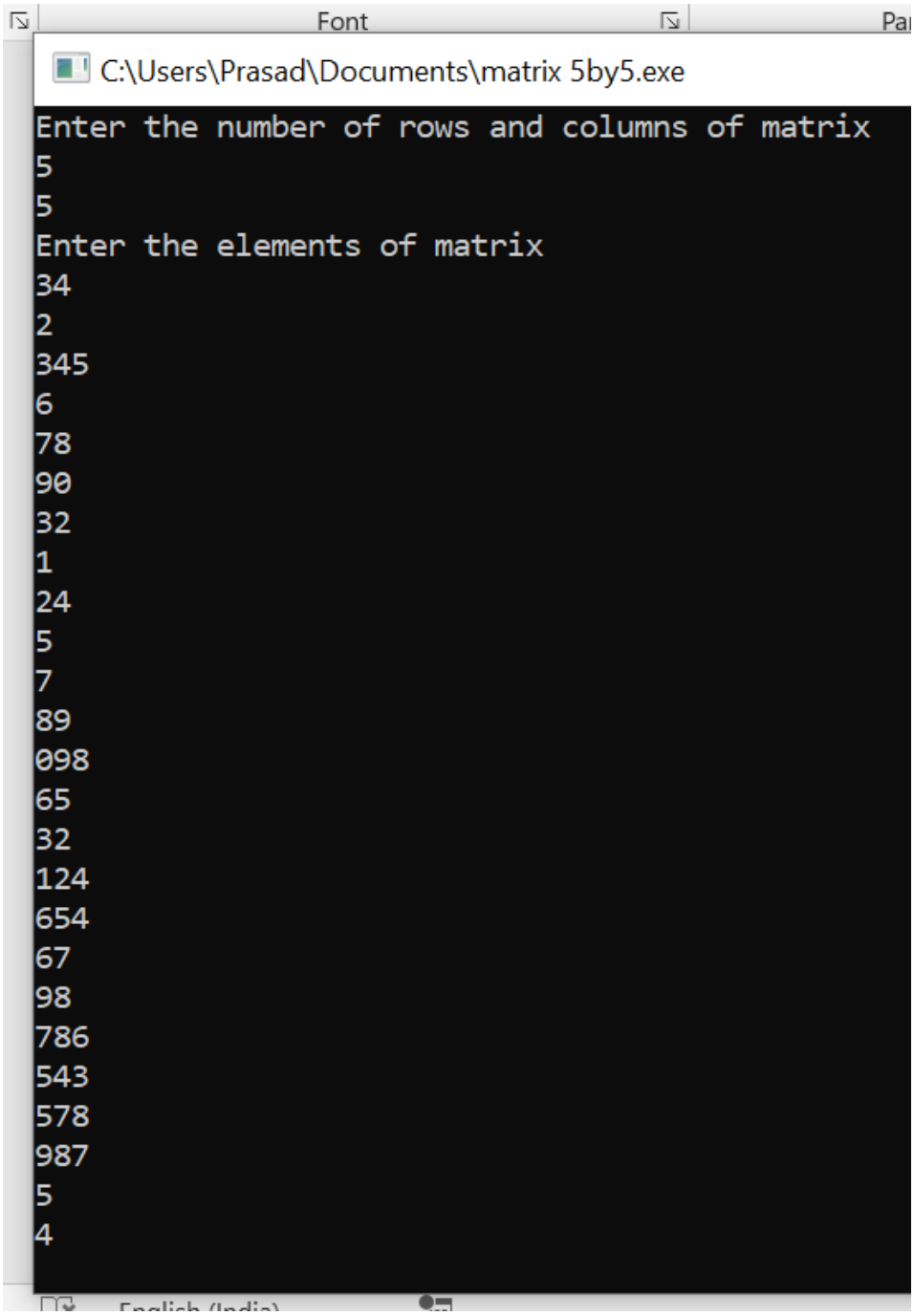

Roll No:BID1900

```
19      {
20          for(d=0;d< n;d++)
21          {
22              printf("%d\t",matrix[c][d]);
23          }
24          printf("\n");
25      }
26      maximum = matrix[0][0];
27      for( c = 0 ; c < m ; c++ )
28      {
29        for( d = 0 ; d < n ; d++ )
30        {
31           if ( matrix[c][d] > maximum )
32              maximum = matrix[c][d];
33        }
34      }
```

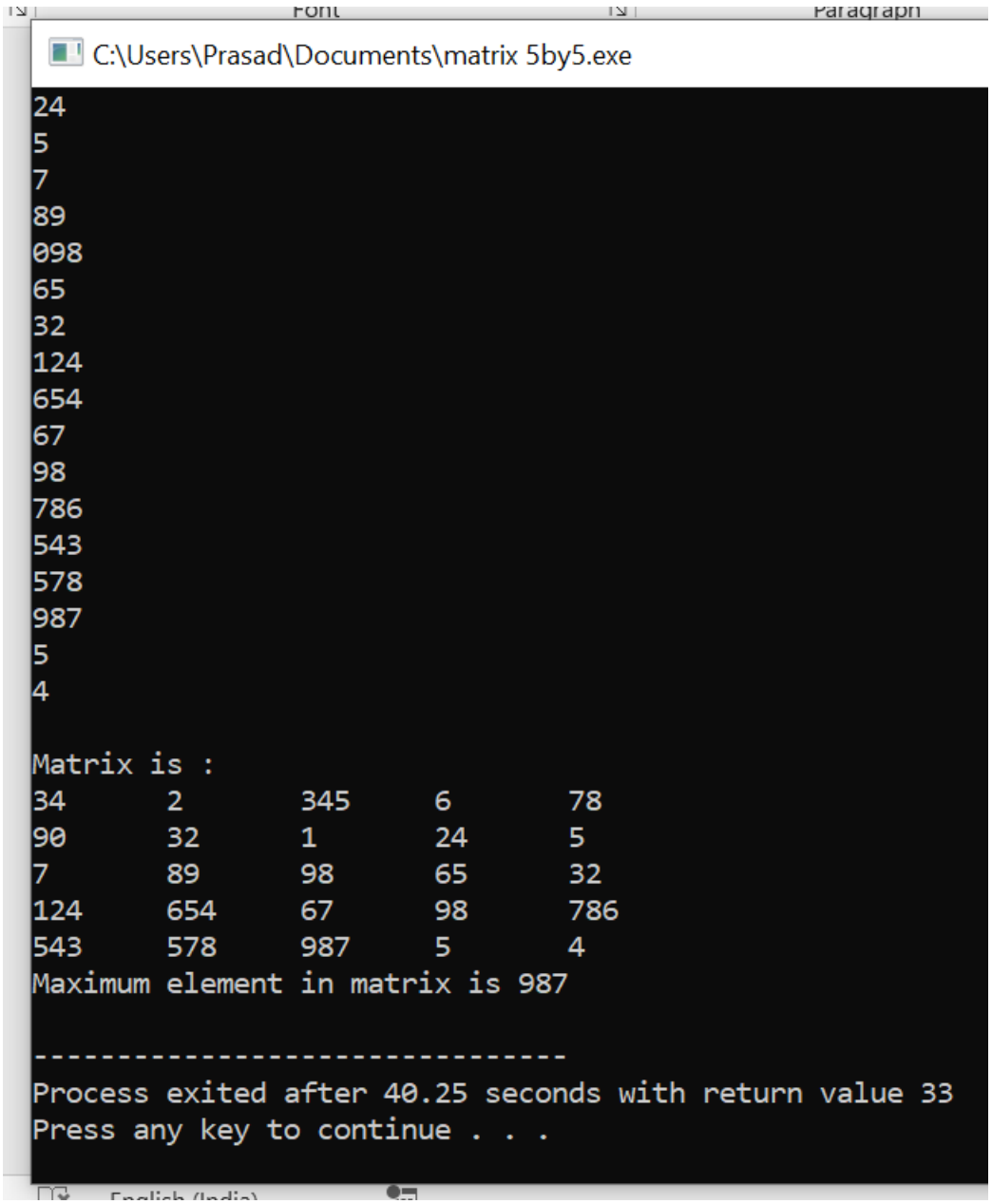
```
34      }
35      printf("Maximum element in matrix is %d\n", maximum);
36      return 0;
37  }
```


Roll No:BID1900

Output:



Roll No:BID1900



Roll No:BID1900

Date:21/9/2020

Exercise 27: Perform a 3 x 3 matrix addition

Algorithm:

Step 1: Start
Step 2: Declaring the array of size 3*3 and other variables
Step 3: Taking Input of array elements using nested loops
Step 4: Calculate sum of two matrices using nested lops
Step 5: PRINT the sum
Step 6: Stop

C code:

Untitled1 | 3 by 3 matrix.c

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

int main()
 {
    int a[10][10], transpose[10][10], r, c, i, j;
    printf("Enter rows and columns: ");
    scanf("%d %d", &r, &c);

    // Assigning elements to the matrix
    printf("\nEnter matrix elements:\n");
    for (i = 0; i < r; ++i)
        for (j = 0; j < c; ++j) {
            printf("Enter element a%d%d: ", i + 1, j + 1);
            scanf("%d", &a[i][j]);
        }

    // Displaying the matrix a[][]
    printf("\nEntered matrix: \n");
```


Roll No:BID1900

Untitled1 | 3 by 3 matrix.c

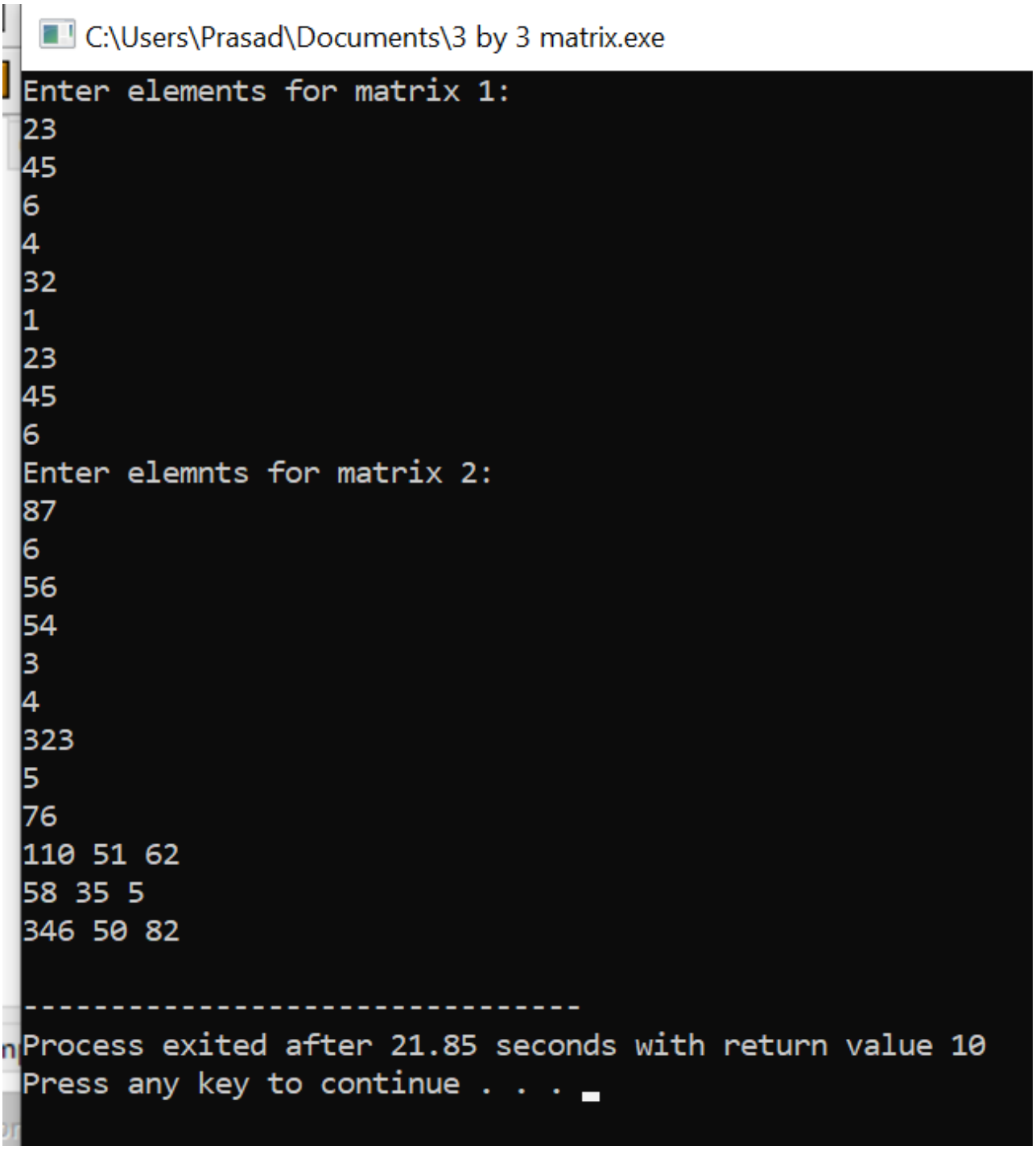
```
16          }
17
18          // Displaying the matrix a[][]
19          printf("\nEntered matrix: \n");
20          for (i = 0; i < r; ++i)
21              for (j = 0; j < c; ++j) {
22                  printf("%d  ", a[i][j]);
23                  if (j == c - 1)
24                      printf("\n");
25              }
26
27          // Finding the transpose of matrix a
28          for (i = 0; i < r; ++i)
29              for (j = 0; j < c; ++j) {
30                  transpose[j][i] = a[i][j];
31              }
32
33          // Displaying the transpose of matrix a
34          printf("\nTranspose of the matrix:\n");
```

Untitled1 | 3 by 3 matrix.c

```
24                      printf("\n");
25              }
26
27          // Finding the transpose of matrix a
28          for (i = 0; i < r; ++i)
29              for (j = 0; j < c; ++j) {
30                  transpose[j][i] = a[i][j];
31              }
32
33          // Displaying the transpose of matrix a
34          printf("\nTranspose of the matrix:\n");
35          for (i = 0; i < c; ++i)
36              for (j = 0; j < r; ++j) {
37                  printf("%d  ", transpose[i][j]);
38                  if (j == r - 1)
39                      printf("\n");
40              }
41          return 0;
42      }
```


Roll No:BID1900

Output:



Roll No:BID1900

Date:21/9/2020

Exercise 28: Perform a 3 x 3 matrix multiplication

Algorithm:

Step 1: Start
Step 2: Declaring the array of size 3*3 and other variables
Step 3: Taking Input of array elements using nested loops
Step 4: Calculate multiplication of two matrices using nested loops
Step 5: PRINT the multiplication
Step 6: Stop

C code:

Untitled1 | 3 by 3 matrix.c | 3 into 3 multiply.c

```
1  #include<stdio.h>
2  #include<conio.h>
3  void main()
4  {
5  int i,j,k,a1[3][3],a2[3][3],mul[3][3];
6  printf("Enter elements for matrix 1:\n");
7  for(i=0;i<=2;i++)
8  {
9   for(j=0;j<=2;j++)
10  {
11   scanf("%d",&a1[i][j]);
12  }
13 }
14 printf("Enter elemnts for matrix 2:\n");
15 for(i=0;i<=2;i++)
16 {
17  for(j=0;j<=2;j++)
18  {
19   scanf("%d",&a2[i][j]);
```


Roll No:BID1900

Untitled1 | 3 by 3 matrix.c | 3 into 3 multiply.c

```
14  printf("Enter elemnts for matrix 2:\n");
15  for(i=0;i<=2;i++)
16  {
17   for(j=0;j<=2;j++)
18   {
19    scanf("%d",&a2[i][j]);
20   }
21  }
22  for(i=0;i<=2;i++)
23  {
24   for(j=0,k=0;j<=2;j++)
25   {
26    mul[i][j]=(a1[i][j]*a2[i][k])+(a1[i][j]*a2[i][k+1])+(a1[i][j]*a2[i][k+2]);
27    printf("%d ",mul[i][j]);
28   }
29  printf("\n");
30  }
31  }
```

Output:

```
C:\Users\Prasad\Documents\3 into 3 multiply.exe
Enter elements for matrix 1:
23
2
1
2
3
4
5
6
7
Enter elemnts for matrix 2:
6
5
3
4
5
6
6
54
4
322 28 14
30 45 60
320 384 448

--------------------------------
Process exited after 13.54 seconds with return value 10
Press any key to continue . . .
```


Roll No:BID1900

Date:21/9/2020

Exercise 29: Find the transpose of a 3x3 matrix

Algorithm:

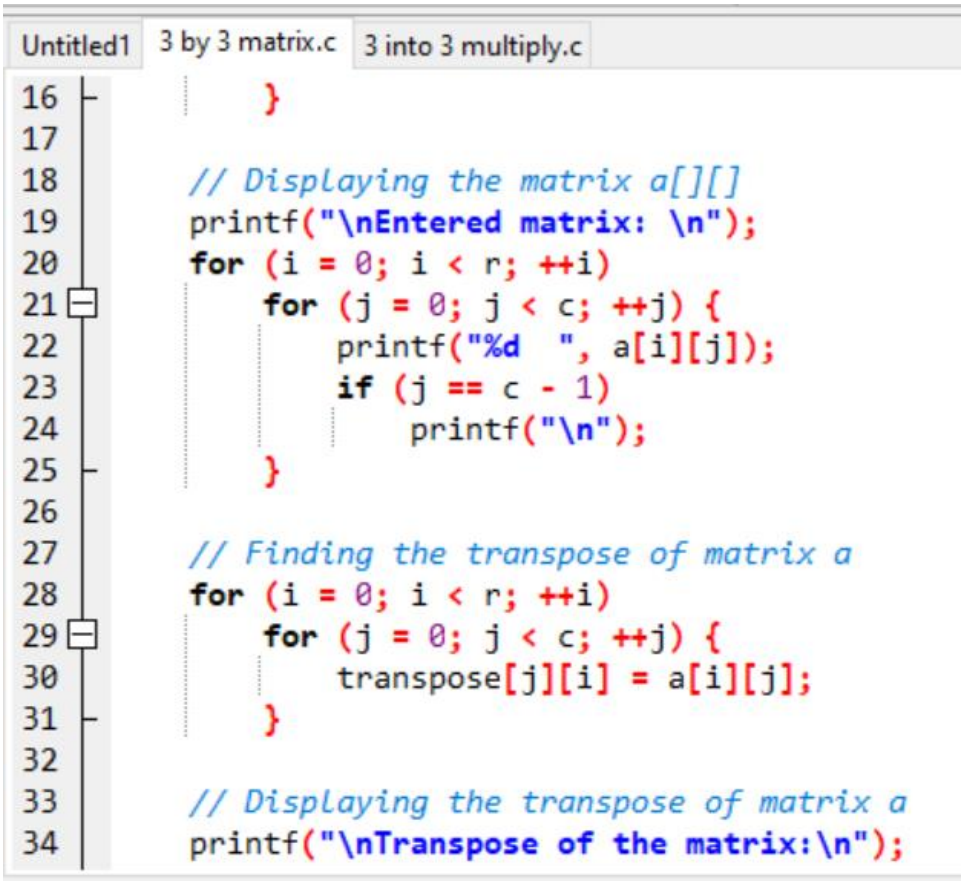
Step 1: Start
Step 2: Declaring the array of size and other variables
Step 3: Assign elements to the matrix
Step 4: Display the matrix
Step 5: Find the transpose of matrix a
Step 6: Display the transpose of matrix a
Step 7: Stop

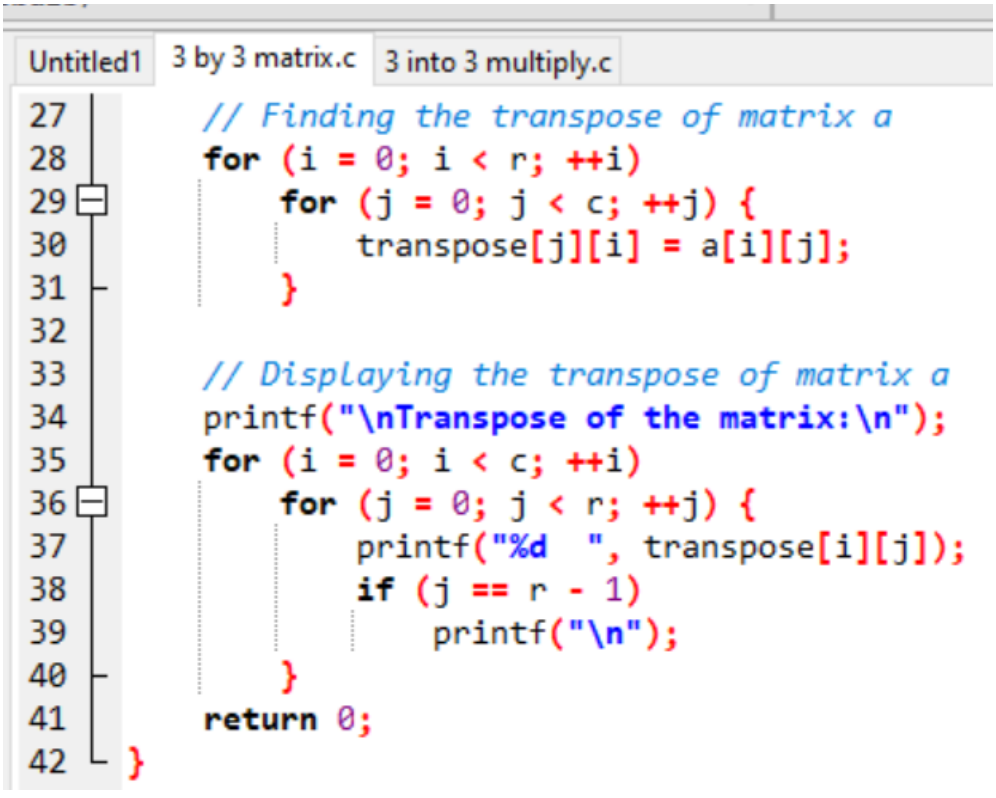
C code:

Untitled1 | 3 by 3 matrix.c | 3 into 3 multiply.c

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

int main()
 {
    int a[10][10], transpose[10][10], r, c, i, j;
    printf("Enter rows and columns: ");
    scanf("%d %d", &r, &c);

    // Assigning elements to the matrix
    printf("\nEnter matrix elements:\n");
    for (i = 0; i < r; ++i)
        for (j = 0; j < c; ++j) {
            printf("Enter element a%d%d: ", i + 1, j + 1);
            scanf("%d", &a[i][j]);
        }

    // Displaying the matrix a[][]
    printf("\nEntered matrix: \n");
```

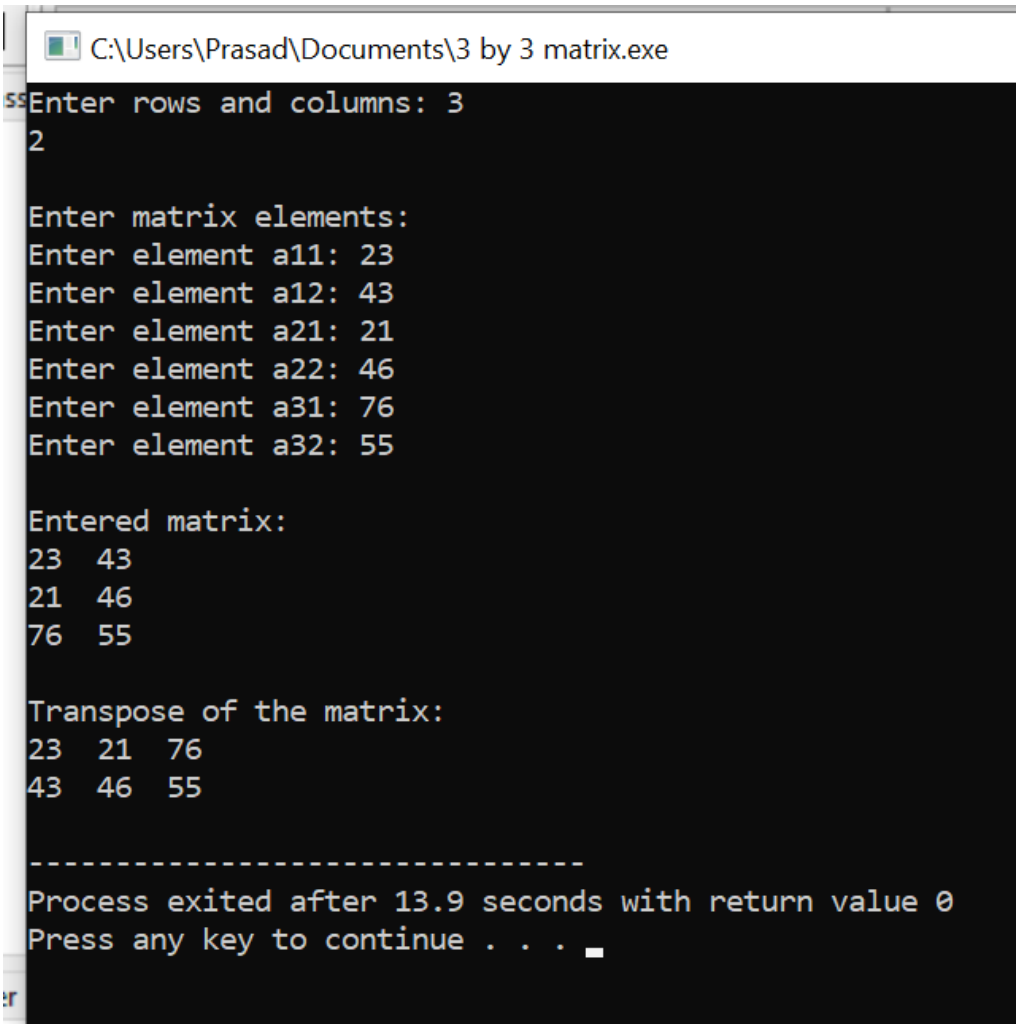

Roll No:BID1900





Roll No:BID1900

Output:



Roll No:BID1900

Date:22/9/2020

Exercise 30: Write a program to extract part of the given string from the specified position

Algorithm:

Step 1: Start
Step 2: Declaring the String of maximum array size 500 and other variables
Step 3: Taking Input of string elements : Specific Position
Step 4: Taking Length of string as input
Step 5: Printing string from n till return value
Step 6: Stop

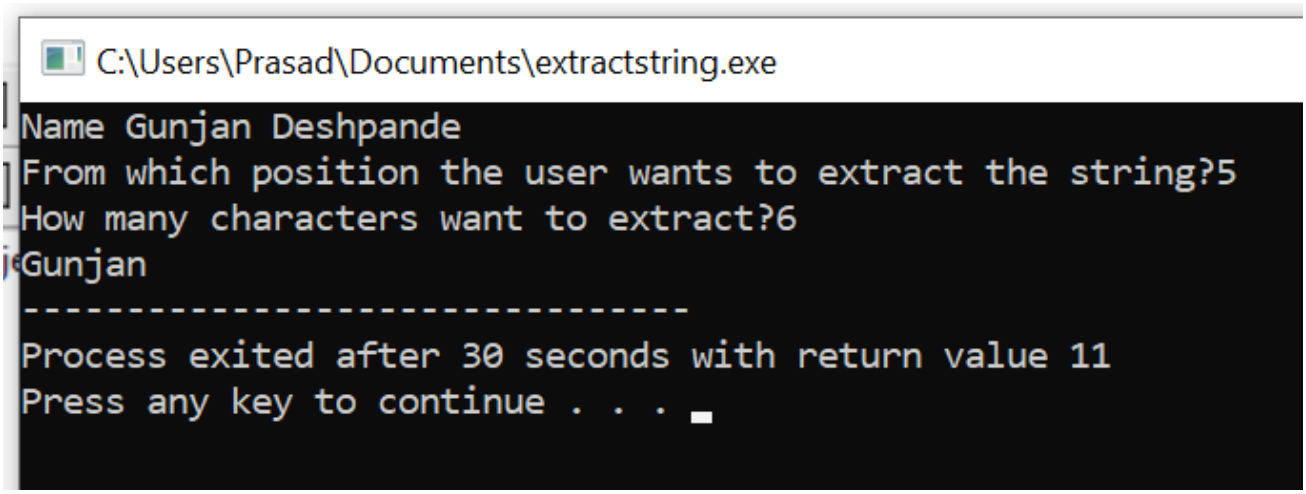
C code:

Untitled1 | 3 by 3 matrix.c | 3 into 3 multiply.c | extractstring.c

```c
#include <stdio.h>
#include <string.h>
void main()
{
    char a[500]; //Declaring the String of maximum array size 500 and other variables
    int n,x,i;

    gets(a);
    printf("From which position the user wants to extract the string?");
    //Taking Input of string elements : Specific Position
    scanf("%d",&n);
    printf("How many characters want to extract?");
    //Taking Length of string as input
    scanf("%d",&x);
    for(i=n;i<=n+x;i++)
        printf("%c",a[i]);
        //Printing string from n till return value
}
```


Roll No:BID1900

Output:



Roll No:BID1900

Date:22/9/2020

Exercise 31: Write a program to sort a set of names stored in an array in alphabetical order

Algorithm:

Step 1: Start

Step 2: Declaring the String of maximum array size 10*10 and other variables

Step 3: Taking names as input from the user

Step 4: By using nested loop and conditional operators : name sorting

Step 5:Print the names in order

Step 6: Stop

C code:

Untitled1 | 3 by 3 matrix.c | 3 into 3 multiply.c | extractstring.c | sort names.c

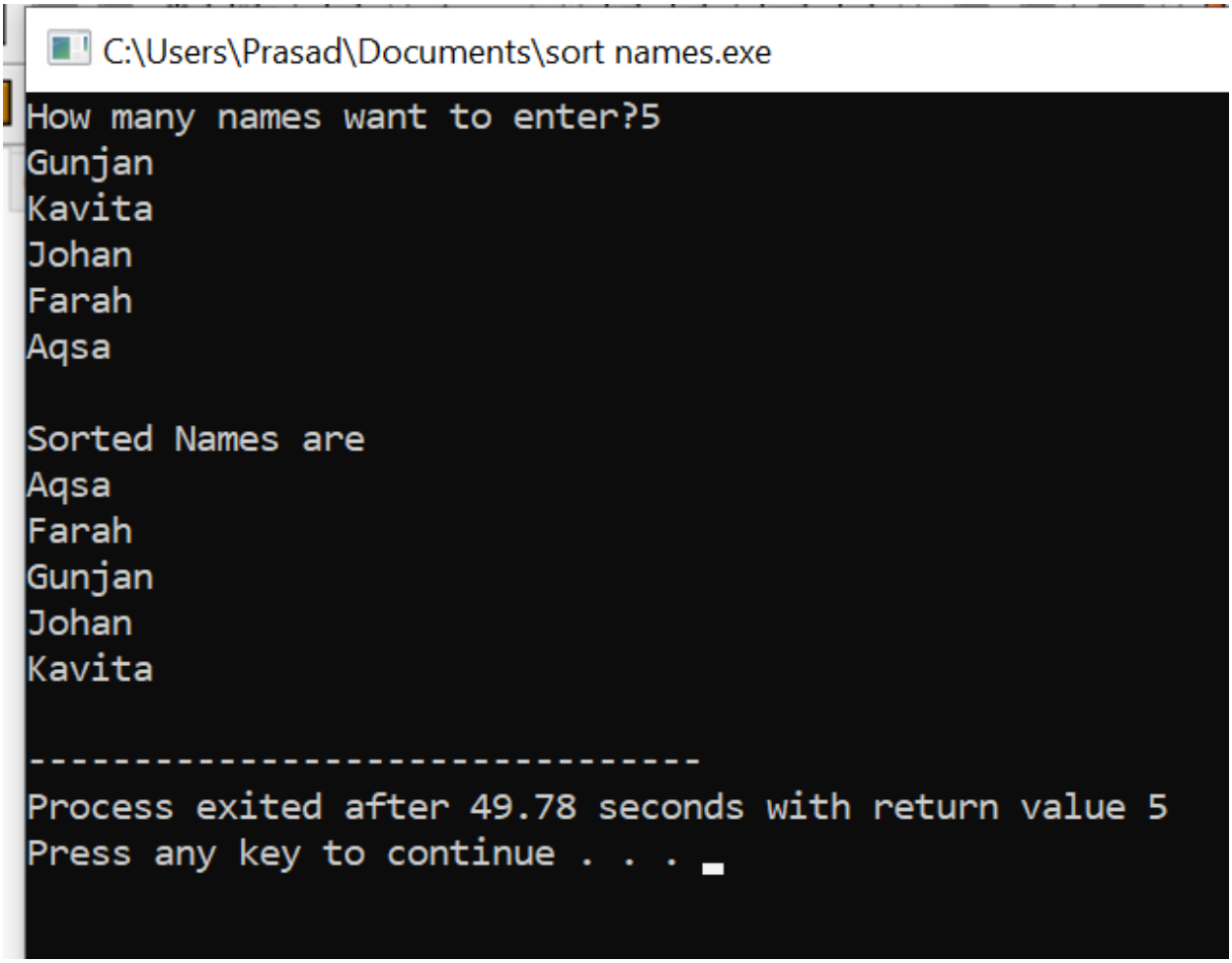
```c
#include <stdio.h>
#include <string.h>
void main()
{
    char names[10][10];  //Declaring the String of maximum array size 10*10 and other variables
    char t[10];
    int i,j,x;

    printf("How many names want to enter?");  //Taking names as input from the user
    scanf("%d",&x);
    //By using nested loop and conditional operators : name sorting
    for(i=0;i<x;i++)
    {
        scanf("%s",&names[i]);
    }
    for(i=0;i<x;i++)
    {
        for(j=i+1;j<x;j++)
        {
```

Untitled1 | 3 by 3 matrix.c | 3 into 3 multiply.c | extractstring.c | sort names.c

```c
    for(i=0;i<x;i++)
    {
        for(j=i+1;j<x;j++)
        {
            if(strcmp(names[i],names[j])>0)
            {
             strcpy(t ,names[i]);
             strcpy(names[i],names[j]);
             strcpy(names[j],t);
            }
        }
    }
    printf("\nSorted Names are \n");
//Print the names in order
    for(i=0;i<x;i++)
        printf("%s\n",names[i]);

}
```


Roll No:BID1900

Output:



Roll No:BID1900

Date:22/9/2020

Exercise 32: Write a program to delete all vowels from a sentence. Assume that the sentence is not more than 80 characterslong.

Algorithm:

Step 1: Start

Step 2: Declaring the String of maximum array size 80 and other variables

Step 3: Take input of String

Step 4:By using nested loop ,conditional operators and logical operators : delete vowels

Step 5: Print the sentence without the vowels

Step 6: Stop

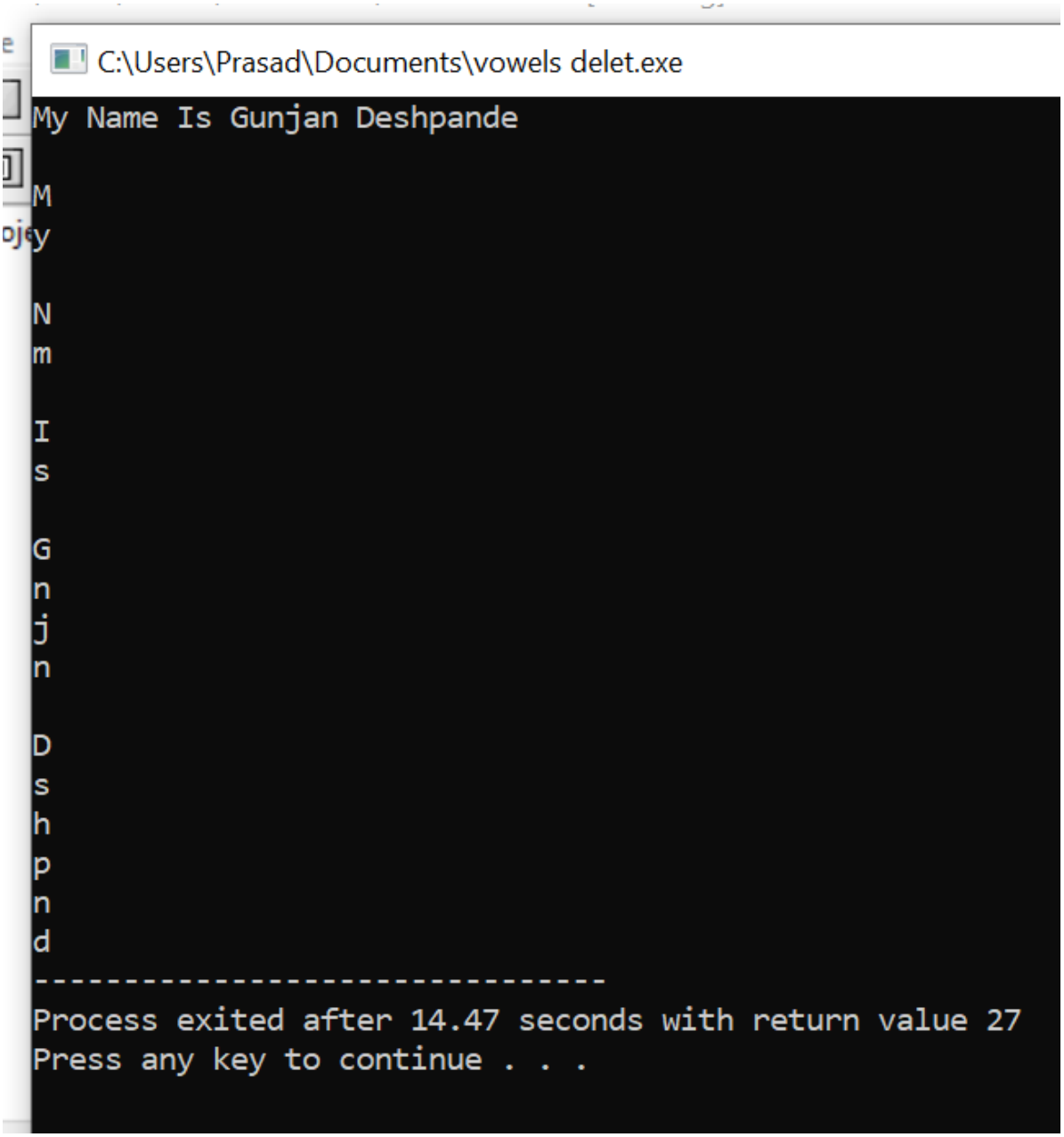
C code:

Untitled1 | 3 by 3 matrix.c | 3 into 3 multiply.c | extractstring.c | sort names.c | [*] vowels delet.c

```c
#include <stdio.h>
#include <string.h>
void main()
{
    char a[80];    //Declaring the String of maximum array size 80 and other variables
    int i;
    gets(a);   //Take input of String
    for(i=0;i<strlen(a);i++)
    {
        //By using for loop ,conditional operators and logical operators : delete vowels
        if(a[i]=='a' || a[i]=='e' || a[i]=='i' || a[i]=='o' || a[i]=='u')
            continue;
        else
        printf("\n%c",a[i]) ; //Print the sentence without the vowels
    }
}
```


Roll No:BID1900

Output:



Roll No:BID1900

Date:22/9/2020

Exercise 33: Create a storage structure for a school library and access the elements

Algorithm:

Step 1: Start

Step 2: Declaring the functions to add book, display book, display book author , to sort, to link

Step 3: In main function , declare library as structure, declare arrays for char book-title[20], char author-name[20] and other variables

Step 4: Display different options

Step 5: Take option as choice from user using switch case

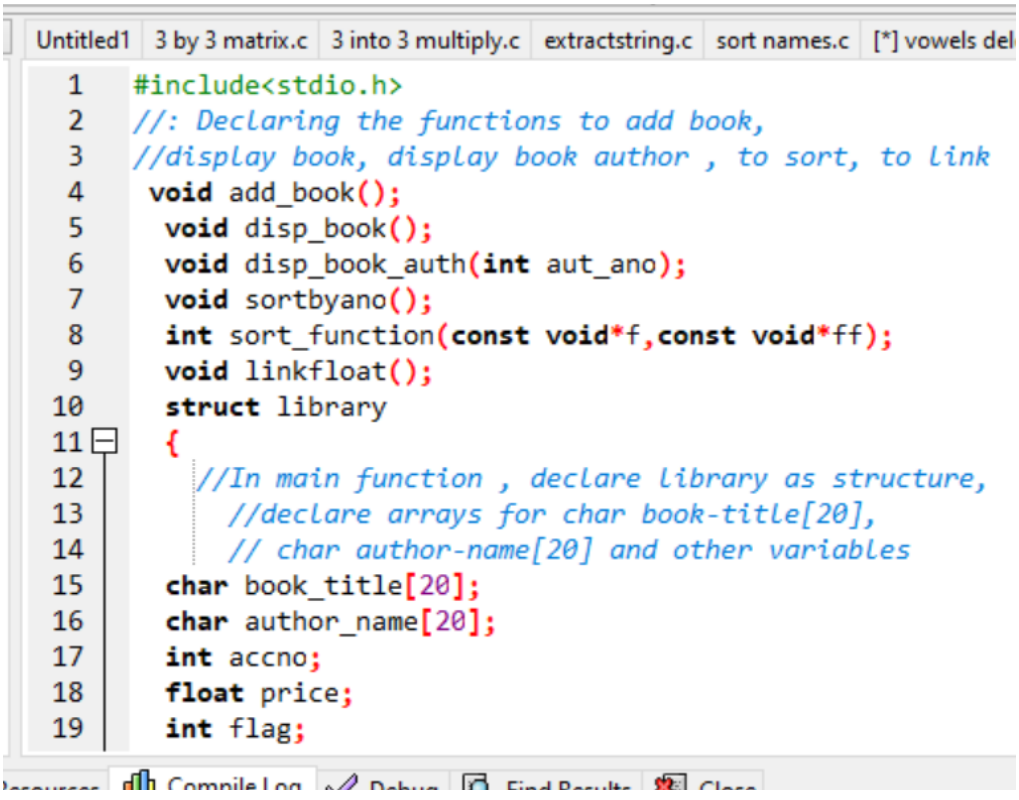
Step 6: Define functions after the main function

Step 7: By using nested loop ,conditional operators define all the functions

Step 8: Print the results

Step 9: Stop

C code:



Roll No:BID1900

Untitled1 | 3 by 3 matrix.c | 3 into 3 multiply.c | extractstring.c | sort names.c | [*] vowels delet.c | [*]

```
19      int flag;
20      };
21      int count;
22      struct library book[10];
23      int main()
24      {
25      int choice;
26    //: Display different options
27      while(1)
28      {
29      printf("What do you want to do?\n");
30      printf("1.Add about a book\n");
31      printf("2.Display book information\n");
32      printf("3.List all the book of given author\n");
33      printf("4.List the title of the specified book\n");
34      printf("5.List the count of book in library\n");
35      printf("6.List the book in order of accession number\n");
36      printf("7.Exit\n");
37      scanf("%d",&choice);
```

Untitled1 | 3 by 3 matrix.c | 3 into 3 multiply.c | extractstring.c | sort names.c | [*] vowels delet.c | [*]

```
37      scanf("%d",&choice);
38      switch(choice)
39      {
40        //: Take option as choice from user using switch case
41      case 1:
42      add_book();
43      break;
44      case 2:
45      disp_book();
46      break;
47      case 3:
48      disp_book_auth(0);
49      break;
50      case 4:
51      disp_book_auth(1);
52      break;
53      case 5:
54      printf("Total number of book in library is %d\n",count);
55      break;
```


Roll No:BID1900

Untitled1 | 3 by 3 matrix.c | 3 into 3 multiply.c | extractstring.c | sort names.c | [*] vowels delet.c

```
49      break;
50      case 4:
51      disp_book_auth(1);
52      break;
53      case 5:
54      printf("Total number of book in library is %d\n",count);
55      break;
56      case 6:
57      sortbyano();
58      break;
59      case 7:
60      exit(0);
61      }
62      }
63      }
64      void add_book()
65      {
66      if(count==9)
67      {
```

Untitled1 | 3 by 3 matrix.c | 3 into 3 multiply.c | extractstring.c | sort names.c | [*] vowels

```
64      void add_book()
65      {
66      if(count==9)
67      {
68      printf("No more space\n");
69      return;
70      }
71       printf("Enter detail of the book\n");
72       printf("Name of the book\n");
73       scanf("%s",book[count].book_title);
74       printf("Name of author\n");
75       scanf("%s",book[count].author_name);
76       printf("Accession number of book\n");
77       scanf("%d",&book[count].accno);
78       printf("Price of the book\n");
79       scanf("%f",book[count].price);
80       printf("Issued\n not issued(0/1):");
81       scanf("%d",&book[count].flag);
82       if((book[count].flag!=0)&&(book[count].flag!=1))
```

Compile Log | Debug | Find Results | Close

Roll No:BID1900

Untitled1 | 3 by 3 matrix.c | 3 into 3 multiply.c | extractstring.c | sort names.c | [*] vowels delet.c

```
82      if((book[count].flag!=0)&&(book[count].flag!=1))
83      {
84      printf("Improper status\n");
85      return;
86      }
87      count++;
88      printf("\n Book details entered\n");
89      }
90      void disp_book()
91      {
92      int i;
93      printf("Details of %d books in library are:\n",count);
94      for(i=0;i<count;i++)
95      {
96      printf("Name of the book:%s\n",book[i].book_title);
97      printf("Name of author:%s\n",book[i].author_name);
98      printf("Accession number of book:%d\n",book[i].accno);
99      printf("Price of the book:%f\n",book[i].price);
100     printf("Status of the book:");
```

Untitled1 | 3 by 3 matrix.c | 3 into 3 multiply.c | extractstring.c | sort names.c | [*] vowels delet.c | [*] library.c

```
99      printf("Price of the book:%f\n",book[i].price);
100     printf("Status of the book:");
101     book[i].flag==0? printf("Issued"):printf("Available");
102     printf("\n\n");
103     }
104     }
105     void disp_book_auth(int aut_ano)
106     {
107     char*nm[20];
108     int accno;
109     int i=0;
110     int dec=0;
111     if(aut_ano==0)
112     {
113     printf("\nEnter author name\n");
114     scanf("%s",&nm);
115     printf("\nDetails of book by author %s in the library are:",nm);
116     }
117     else
```


Roll No:BID1900

Untitled1 | 3 by 3 matrix.c | 3 into 3 multiply.c | extractstring.c | sort names.c | [*] vowels delet.c | [*] library.c

```
114     scanf("%s",&nm);
115     printf("\nDetails of book by author %s in the library are:",nm)
116     }
117     else
118     {
119     printf("\nEnter accession number opf the book\n");
120     scanf("%d",&accno);
121     printf("\nDetails of the books with accession no.%d:",accno);
122     }
123     for(i=0;i<count;i++)
124     {
125      if((strcmp(nm,book[i].author_name)==0)&&(aut_ano==0));
126      dec++;
127     {
128     if(aut_ano==1)
129     {
130     if(book[i].accno==accno)
131     dec++;
132     else
```

Untitled1 | 3 by 3 matrix.c | 3 into 3 multiply.c | extractstring.c | sort names.c | [*] vowels delet.c

```
132     else
133     continue;
134     }
135     else
136     break;
137     }
138     printf("Name of book:%s\n",book[i].book_title);
139     printf("Name of author:%s\n",book[i].author_name);
140     printf("Accession number:%d\n",book[i].accno);
141     printf("Price of book:%f\n",book[i].price);
142     printf("Status of book:\n");
143     book[i].flag==0? printf("Issused"):printf("Available");
144     printf("\n\n");
145     }
146     if(dec==0)
147     printf("No such book\n");
148     }
149     void sortbyano()
150     {
```

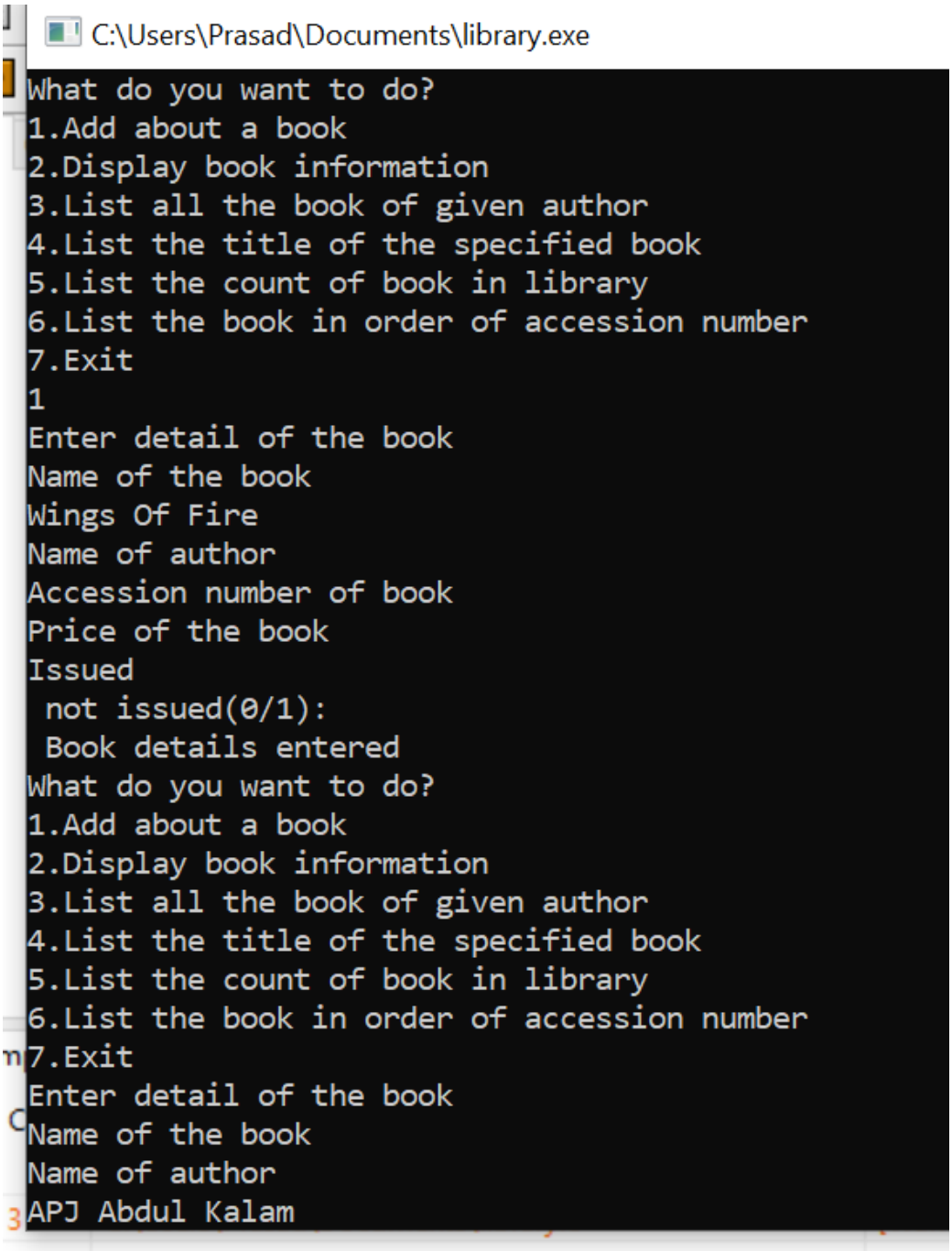

Roll No:BID1900

Untitled1 | 3 by 3 matrix.c | 3 into 3 multiply.c | extractstring.c | sort names.c | [*] vowels delet.c | [*] library.c

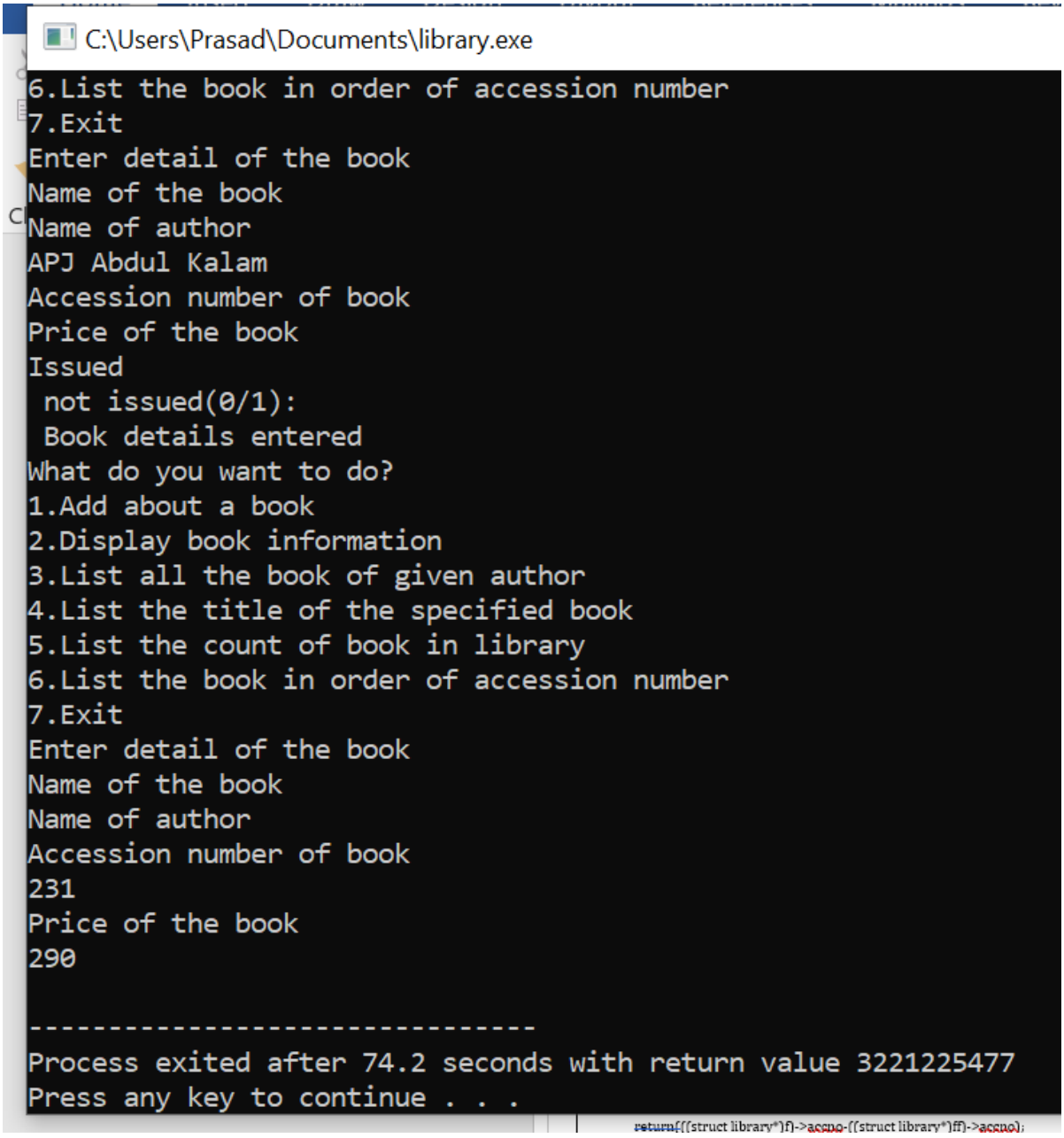
```
147     printf("No such book\n");
148     }
149     void sortbyano()
150     {
151     qsort((struct library*)book,count,sizeof(book[0]),sort_function);
152     printf("After sortig the accession:\n");
153     disp_book();
154     }
155     int sort_function(const void*f,const void*ff)
156     {
157     return(((struct library*)f)->accno-((struct library*)ff)->accno);
158     }
159     void linkfloat()
160     {
161     float a=0,*b;
162     b=&a;
163     a=*b;
164     }
165
```


Roll No:BID1900

Output:



Roll No:BID1900



Roll No:BID1900

Date:23/9/2020

Exercise 34: Read a file and display its contents along with line number before each line

Algorithm:

Step 1: Start

Step 2: Declare files using pointer and other variables

Step 3: Count for line number

Step 4: Opening the file in read only format: fs=fopen("36.c","r");

Step 5: By using conditional operators check whether If file doesn't exist, then print cannot open source file

Step 6: By using loop read each character: use conditional operator ,if(ch==EOF), then break, else if(ch=='\n'), new line encountered

Step 7: Printing the character

Step 8: Close the file if opened

Step 9: Stop

C code:

[*] content copy .c | line num.c

```c
#include<stdio.h>
main()
{
//Declare files using pointer and other variables
    FILE *fs;
    char ch;
    int i=1;
    //Opening the file in read only format: fs=fopen("36.c","r");
    fs=fopen("36.c","r");
    if(fs==NULL)
    {
        printf("can't open source file");
        exit(1);
    }
    //By using conditional operators check whether
    //If file doesn't exist, then print cannot open source file
    printf("%d",i++);
    //By using loop read each character:
    //use conditional operator , if(ch==EOF), then break, else if(ch=='\n'), new line encountered
```


Roll No:BID1900

[*] content copy .c | line num.c

```c
    //use conditional operator , if(ch==EOF), then break, else if(ch=='\n'), new line encountered
    while(1)
    {
        ch=fgetc(fs);
        if(ch==EOF)
        {
            break;
        }
        printf("%c",ch);
        //Printing the character
        if(ch=='\n')
        {
            printf("%d",i);
            i++;
        }
    }
    // Close the file if opened
    fclose(fs);
}
```

Output:

C:\Users\Prasad\Documents\line num.exe

```c
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
void main()
{
      FILE *fptr1,*fptr2;
      char filename[100],c;

      printf("enter the file name \n");
      scanf("%s",filename);

              fptr1=fopen(filename,"r");
              if(fptr1==NULL)
              {
                      printf("cannot open file %s \n", filename);
                      exit(0);
              }
      printf("enter the filename \n");
   scanf ("%s",filename);
              fptr2=fopen(filename,"w");
              if(fptr2==NULL)
              {
                      printf("cannot open file%s\n",filename);
                      exit(0);
              }
              c=fgetc(fptr1);
              while(c!=EOF)
              {
                      fputc(c,fptr2);
```


Roll No:BID1900

Date:23/9/2020

Exercise 35: Program to append the contents of one file at the end of another file

Algorithm:

Step 1: Start

Step 2: Declare files using pointer and other variables

Step 3: Count for line number

Step 4: Opening the file in read only formatfp1 = fopen(fname1, "r");

Step 5: By using conditional operators check whether If file doesn't exist, then print cannot open source file

Step 6: By using loop read each character: use conditional operator ,if(ch==EOF), then break, else if(ch=='\n'), new line encountered

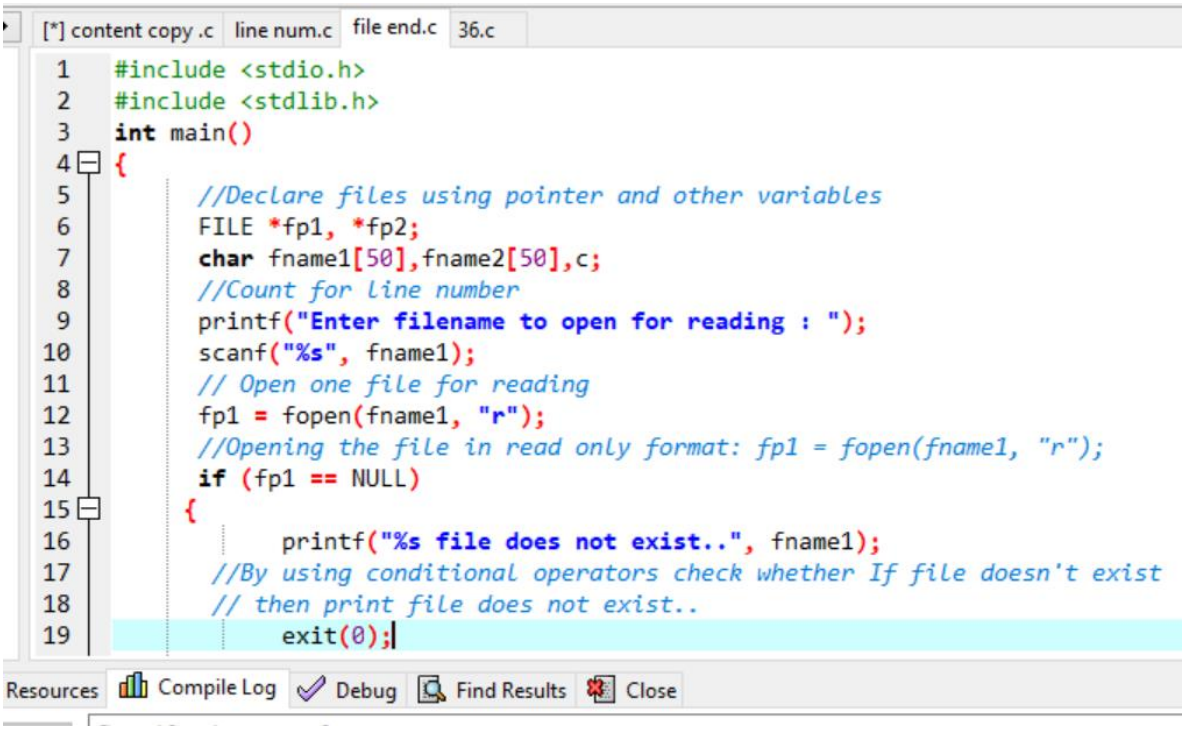
Step 7: Printing the character

Step 8: Repeat the above steps with other files as well

Step 9: Close the files if opened

Step 10: Stop

C code:



Roll No:BID1900

[*] content copy .c | line num.c | file end.c | 36.c

```
19              exit(0);
20          }
21          printf("\nEnter filename to append the content : ");
22          scanf("%s", fname2);
23          // Open another file for appending content
24          //Printing the character
25          //Repeat the above steps with other files as well
26          fp2 = fopen(fname2, "a");
27          if (fp2 == NULL)
28          {
29              printf("%s file does not exist...", fname2);
30              exit(0);
31          }
32          // Read content from file
33          c = fgetc(fp1);
34          while (c != EOF)
35          {
36              fputc(c,fp2);
37              c = fgetc(fp1);
```

```
36              fputc(c,fp2);
37              c = fgetc(fp1);
38          }
39          printf("\nContent in %s appended to %s", fname1,fname2);
40          fclose(fp1);
41          fclose(fp2);
42          return 0;
43      }
44
```

Output:

```
C:\Users\Prasad\Documents\file end.exe
Enter filename to open for reading : 36.c

Enter filename to append the content : line num.c

Content in 36.c appended to line
--------------------------------
Process exited after 12.95 seconds with return value 0
Press any key to continue . . .
```


Roll No:BID1900

Date:23/9/2020

Exercise 36: Program to copy the content of one file to another

Algorithm:

Step 1: Start
Step 2: Declare files and other variables
Step 3: Count for line number
Step 4: Opening the file for reading fptr1 = fopen(filename, "r");
Step 5: By using conditional operators check whether If file doesn't exist, then print cannot open source file
Step 6: By using loop read each character: use conditional operator , while (c != EOF)
Step 7: Opening the file fptr2 = fopen(filename, "w");
Step 8: Copying the contents in 2'nd file using loop and conditional operator.
Step 9: Close the files if opened
Step 10: Stop

C code:

[*] content copy .c

```c
#include <stdio.h>
#include <stdlib.h>
int main()
{
//Declare files and other variables
    FILE *fptr1, *fptr2;
    char filename[100], c;
  //Count for line number
    printf("Enter the filename to open for reading \n");
    scanf("%s", filename);

    // Opening the file for reading fptr1 = fopen(filename, "r");
    fptr1 = fopen(filename, "r");
    if (fptr1 == NULL)
    {
        printf("Cannot open file %s \n", filename);
        exit(0);
    }
    printf("Enter the filename to open for writing \n");
```


Roll No:BID1900

[*] content copy .c

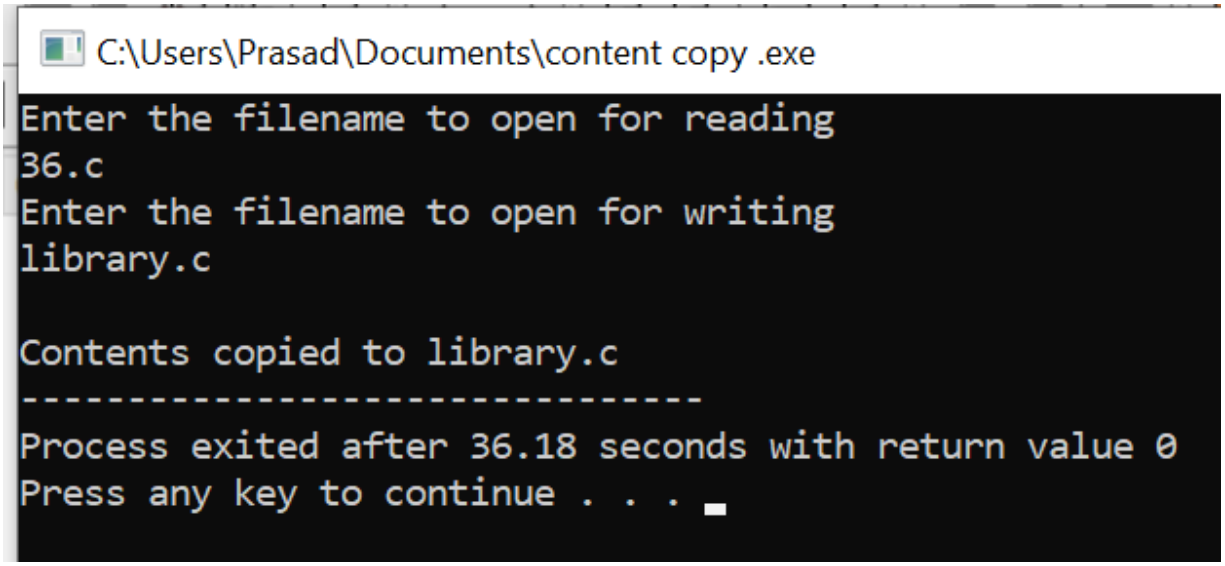
```c
    printf("Enter the filename to open for writing \n");
    scanf("%s", filename);

    // Open another file for writing
    fptr2 = fopen(filename, "w");
    if (fptr2 == NULL)
    {
        printf("Cannot open file %s \n", filename);
        exit(0);
    }
  //By using loop read each character:
  // use operator , while (c != EOF)
    // Read contents from file

    c = fgetc(fptr1);
    while (c != EOF)
    {
        fputc(c, fptr2);
        c = fgetc(fptr1);
```

[*] content copy .c

```c
  //By using loop read each character:
  // use operator , while (c != EOF)
    // Read contents from file

    c = fgetc(fptr1);
    while (c != EOF)
    {
        fputc(c, fptr2);
        c = fgetc(fptr1);
    }
  //Copying the contents in 2'nd file using loop and conditional operator.
    printf("\nContents copied to %s", filename);

    fclose(fptr1);
    fclose(fptr2);
//Close the files if opened
    return 0;
}
```


Roll No:BID1900

Output:



Roll No:BID1900

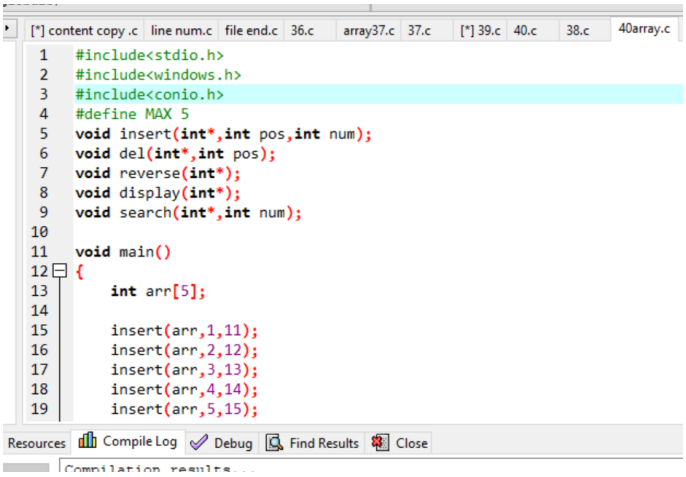
Date:29/9/2020

Exercise 37:Implement the concept of Arrays using Data structures

Algorithm:

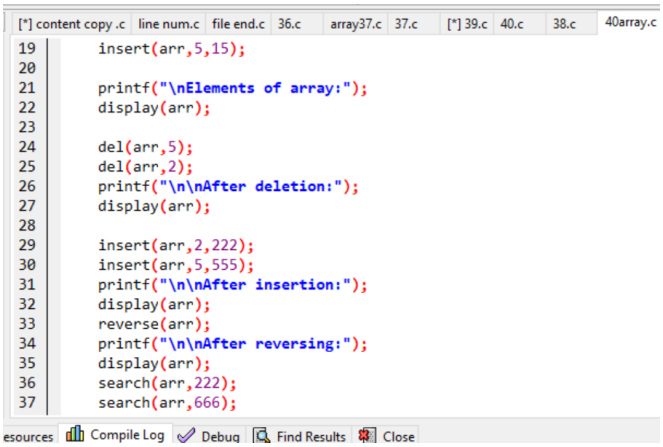
Step 1: Start
Step 2: Declaring the functions to insert ,deletereverse,display,search
Step 3: In main function, declare library as structure, declare arrays of size 5.
Step 4: Call insert function to insert elements in given position of array, then print them
Step 5: Call display function in given position of array
Step 6: Call delete function to delete elements from arrya
Step 7: Call reverse function to print the array in reverse order
Step 8:Call search function to search different elements of array
Step 9: Define functions after the main function
Step 10: By using loop , define all the functions
Step 11: Print the results
Step 12: Stop

C code:

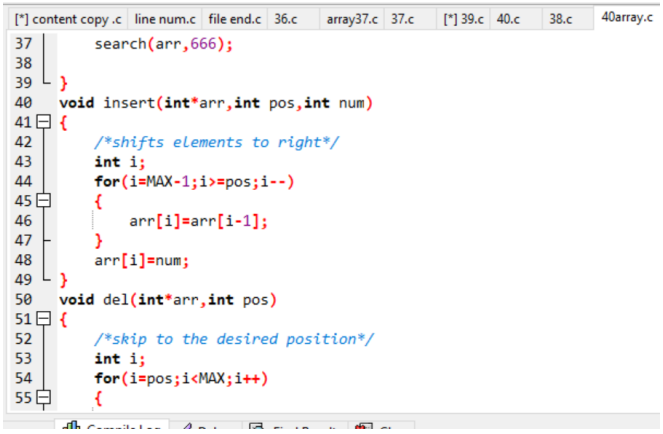



```c
#include<stdio.h>
#include<windows.h>
#include<conio.h>
#define MAX 5
void insert(int*,int pos,int num);
void del(int*,int pos);
void reverse(int*);
void display(int*);
void search(int*,int num);

void main()
{
    int arr[5];

    insert(arr,1,11);
    insert(arr,2,12);
    insert(arr,3,13);
    insert(arr,4,14);
    insert(arr,5,15);
```

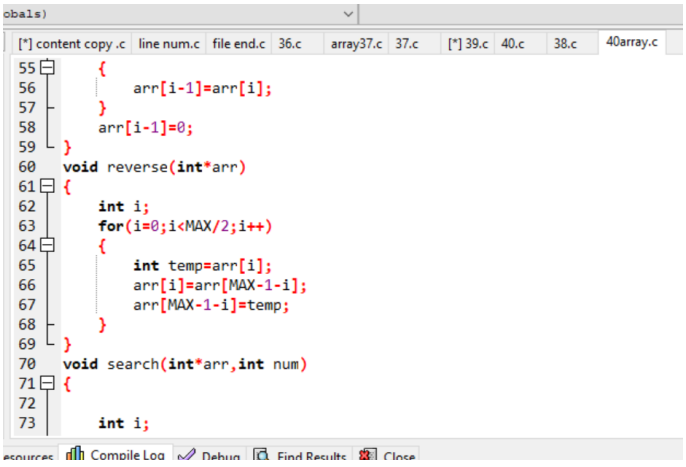

Roll No:BID1900



```
[*] content copy .c | line num.c | file end.c | 36.c | array37.c | 37.c | [*] 39.c | 40.c | 38.c | 40array.c

19      insert(arr,5,15);
20
21      printf("\nElements of array:");
22      display(arr);
23
24      del(arr,5);
25      del(arr,2);
26      printf("\n\nAfter deletion:");
27      display(arr);
28
29      insert(arr,2,222);
30      insert(arr,5,555);
31      printf("\n\nAfter insertion:");
32      display(arr);
33      reverse(arr);
34      printf("\n\nAfter reversing:");
35      display(arr);
36      search(arr,222);
37      search(arr,666);

esources | Compile Log | Debug | Find Results | Close
```



```
[*] content copy .c | line num.c | file end.c | 36.c | array37.c | 37.c | [*] 39.c | 40.c | 38.c | 40array.c

37      search(arr,666);
38
39  }
40  void insert(int*arr,int pos,int num)
41  {
42      /*shifts elements to right*/
43      int i;
44      for(i=MAX-1;i>=pos;i--)
45      {
46          arr[i]=arr[i-1];
47      }
48      arr[i]=num;
49  }
50  void del(int*arr,int pos)
51  {
52      /*skip to the desired position*/
53      int i;
54      for(i=pos;i<MAX;i++)
55      {
```



```
obals)

[*] content copy .c | line num.c | file end.c | 36.c | array37.c | 37.c | [*] 39.c | 40.c | 38.c | 40array.c

55      {
56          arr[i-1]=arr[i];
57      }
58      arr[i-1]=0;
59  }
60  void reverse(int*arr)
61  {
62      int i;
63      for(i=0;i<MAX/2;i++)
64      {
65          int temp=arr[i];
66          arr[i]=arr[MAX-1-i];
67          arr[MAX-1-i]=temp;
68      }
69  }
70  void search(int*arr,int num)
71  {
72
73      int i;

esources | Compile Log | Debug | Find Results | Close
```


Roll No:BID1900

```c
void search(int*arr,int num)
{

    int i;
    for(i=0;i<MAX;i++)
    {
        if(arr[i]==num)
        {
            printf("\n\nThe element %d is present at %dth position.",num,i+1);
            return;
        }
    }
if(i==MAX)
    {
        printf("\n\nThe element %d is not present in the array.",num);
    }
}
void display(int*arr)
{
```

```c
            return;
        }
    }
if(i==MAX)
    {
        printf("\n\nThe element %d is not present in the array.",num);
    }
}
void display(int*arr)
{
    /*traverse the entire array*/
    int i;
    printf("\n");
    for(i=0;i<MAX;i++)
    {
        printf("%d\t",arr[i]);
    }
}
```

Output:

```
C:\Users\Prasad\Documents\40array.exe

Elements of array:
11      12      13      14      15

After deletion:
11      13      14      0       0

After insertion:
11      222     13      14      555

After reversing:
555     14      13      222     11

The element 222 is present at 4th position.

The element 666 is not present in the array.
--------------------------------
Process exited after 0.06537 seconds with return value 46
Press any key to continue . . .
```


Roll No:BID1900

Date:29/9/2020

Exercise 38: Implement the concept of Strings using Data structures

Algorithm:

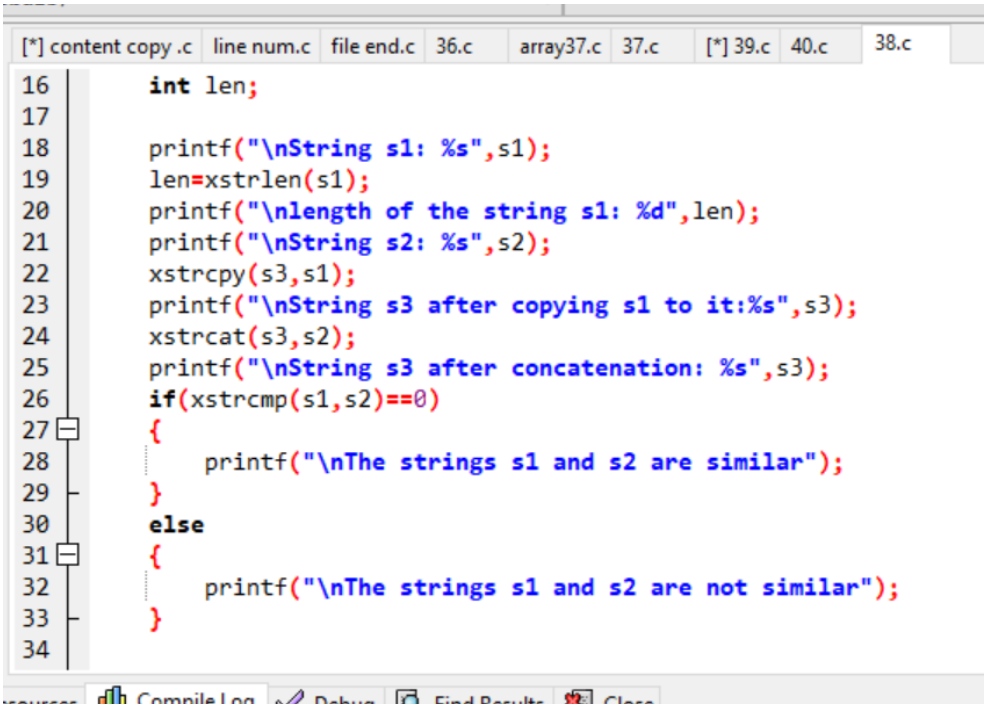
Step 1: Start
Step 2: Declaring the functions for string length, copy ,concatenation , comapre
Step 3: In main function, declare arrays of different sizes.
Step 4: Call length function to finds the length of the string, then print them
Step 5: Call copy function that copies source string s to the target string
Step 6: Call delete function to delete elements from arrya
Step 7: Call concatenation function that Concatenates the two strings
Step 8: Call comaprision function that compares two strings s and t for equality
Step 9: Define functions after the main function
Step 10: By using loop , define all the functions
Step 11: Print the results
Step 12: Stop

C code:

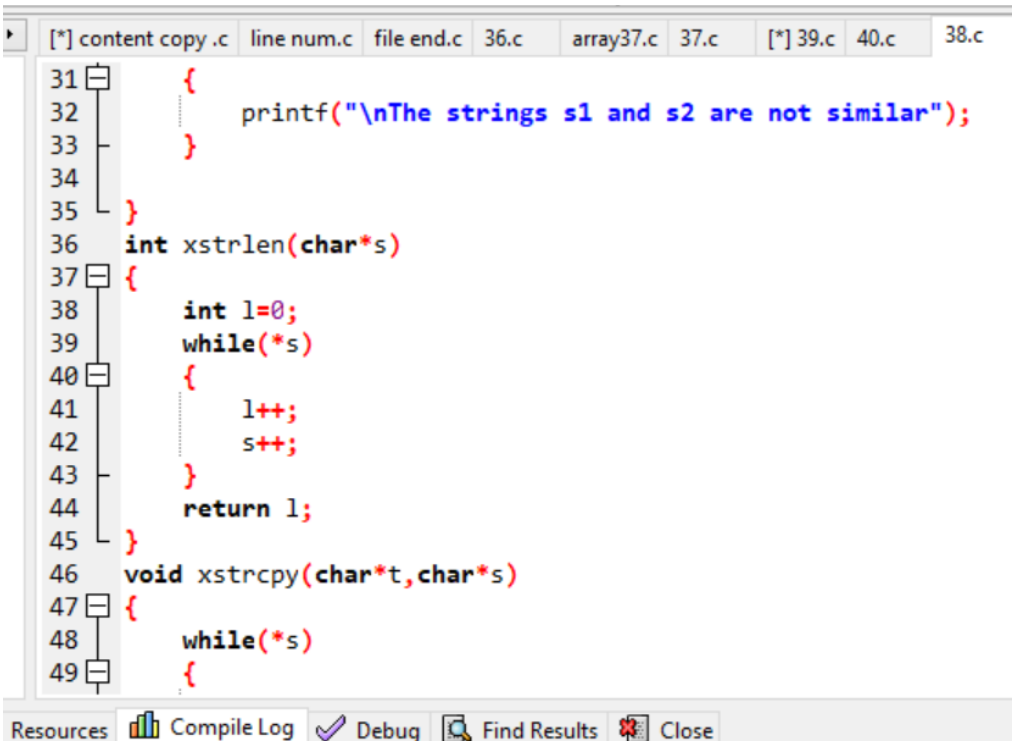
[*] content copy .c | line num.c | file end.c | 36.c | array37.c | 37.c | [*] 39.c | 40.c | 38.c

```c
#include<stdio.h>
#include<conio.h>
#include<string.h>
#include<windows.h>
int xstrlen(char*);
void xstrcpy(char*,char*);
void xstrcat(char*,char*);
int xstrcmp(char*,char*);
void show(char*);

void main()
{
    char s1[]="kicit";
    char s2[]="Nagpur";
    char s3[20];
    int len;

    printf("\nString s1: %s",s1);
    len=xstrlen(s1);
```


Roll No:BID1900



```
[*] content copy .c | line num.c | file end.c | 36.c | array37.c | 37.c | [*] 39.c | 40.c | 38.c

16      int len;
17
18      printf("\nString s1: %s",s1);
19      len=xstrlen(s1);
20      printf("\nlength of the string s1: %d",len);
21      printf("\nString s2: %s",s2);
22      xstrcpy(s3,s1);
23      printf("\nString s3 after copying s1 to it:%s",s3);
24      xstrcat(s3,s2);
25      printf("\nString s3 after concatenation: %s",s3);
26      if(xstrcmp(s1,s2)==0)
27      {
28          printf("\nThe strings s1 and s2 are similar");
29      }
30      else
31      {
32          printf("\nThe strings s1 and s2 are not similar");
33      }
34
```



```
[*] content copy .c | line num.c | file end.c | 36.c | array37.c | 37.c | [*] 39.c | 40.c | 38.c

31      {
32          printf("\nThe strings s1 and s2 are not similar");
33      }
34
35  }
36  int xstrlen(char*s)
37  {
38      int l=0;
39      while(*s)
40      {
41          l++;
42          s++;
43      }
44      return l;
45  }
46  void xstrcpy(char*t,char*s)
47  {
48      while(*s)
49      {

Resources | Compile Log | Debug | Find Results | Close
```

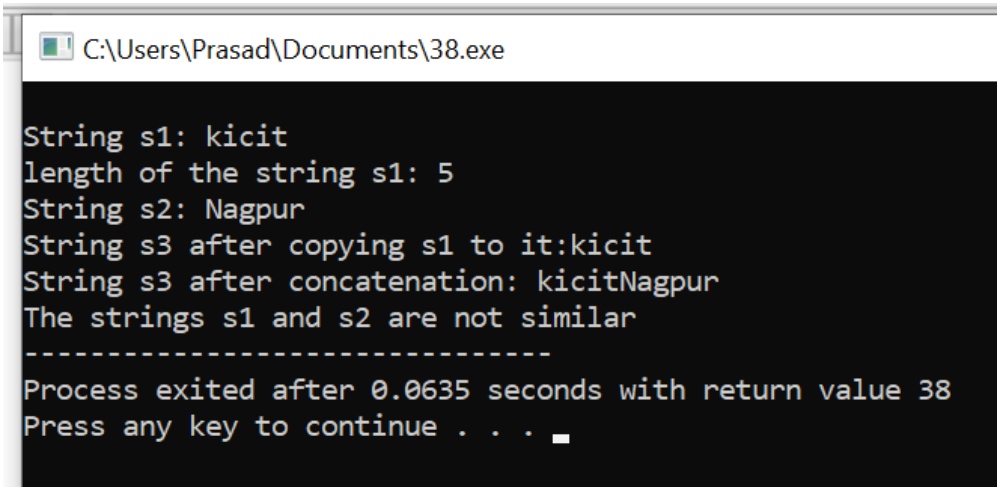

Roll No:BID1900

[*] content copy .c | line num.c | file end.c | 36.c | array37.c | 37.c | [*] 39.c | 40.c | 38.c

```c
void xstrcpy(char*t,char*s)
{
    while(*s)
    {
        *t=*s;
        t++;
        s++;
    }
    *t='\0';
}
void xstrcat(char*t,char*s)
{
    while(*t)
    {
        t++;
    }
    while(*s)
    {
        *t++=*s++;
    }
    *t='\0';
}
int xstrcmp(char*s,char*t)
{
    while(*s==*t)
    {
        if(!(*s))
        {
            return 0;
        }
        s++;
        t++;
    }
    return(*s-*t);
}
```


Roll No:BID1900

Output:



Roll No:BID1900

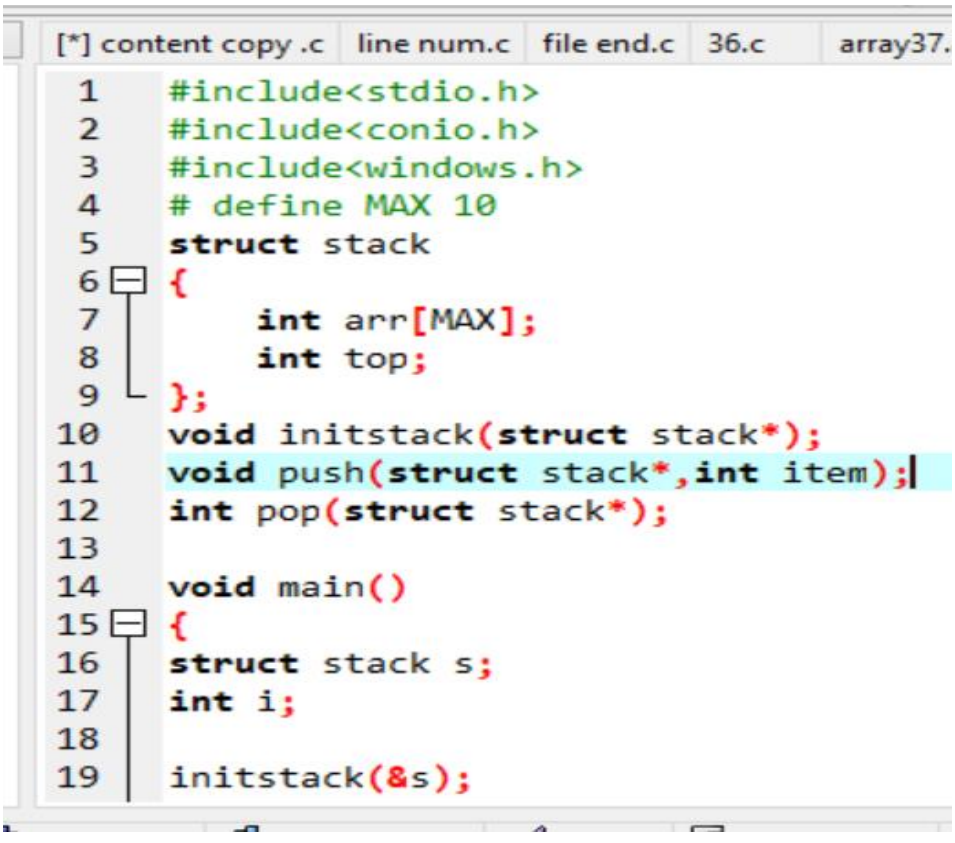
Date:29/9/2020

Exercise 39: Implement the concept of Stack using Data structures

Algorithm:

Step 1: Start
Step 2: Declaring the functions for initstck, push, pop
Step 3: In main function , declare array of size 10.
Step 4: Call initstackfunction to initializes the stack
Step 5: Call push function that copies source string to the target adds an element to the stack
Step 6: Call pop function to delete elements from array
Step 7: Define functions after the main function
Step 8: By conditional operators , define all the functions
Step 9: Print the results
Step 10: Stop

C code:



Roll No:BID1900

```
[*] content copy .c | line num.c | file end.c | 36.c | arra
16  struct stack s;
17  int i;
18
19  initstack(&s);
20  push(&s,11);
21  push(&s,23);
22  push(&s,-8);
23  push(&s,16);
24  push(&s,27);
25  push(&s,14);
26  push(&s,20);
27  push(&s,39);
28  push(&s,2);
29  push(&s,15);
30  push(&s,7);
31
32  i=pop(&s);
33  printf("\n\nItem popped: %d",i);
34  i=pop(&s);
```

Resources | Compile Log | Debug | Find Resu

```
[*] content copy .c | line num.c | file end.c | 36.c | array37.c
34  i=pop(&s);
35  printf("\nItem popped: %d",i);
36  i=pop(&s);
37  printf("\nItem popped: %d",i);
38  i=pop(&s);
39  printf("\nItem popped: %d",i);
40  i=pop(&s);
41  printf("\nItem popped: %d",i);
42
43  }
44  void initstack(struct stack*s)
45  {
46      s->top=-1;
47  }
48  void push(struct stack*s,int item)
49  {
50      if(s->top==MAX-1)
51      {
52          printf("\nStack is full.");
```

Resources | Compile Log | Debug | Find Results

Roll No:BID1900

```
[*] content copy .c | line num.c | file end.c | 36.c | array37.c
52          printf("\nStack is full.");
53          return;
54      }
55      s->top++;
56      s->arr[s->top]=item;
57  }
58  int pop(struct stack*s)
59  {
60      int data;
61      if(s->top==-1)
62      {
63          printf("\nStack is empty.");
64          return NULL;
65      }
66      data=s->arr[s->top];
67      s->top--;
68      return data;
69  }
70
```

Output:

```
C:\Users\Prasad\Documents\39.exe

Stack is full.

Item popped: 15
Item popped: 2
Item popped: 39
Item popped: 20
Item popped: 14
------------------------------
Process exited after 0.06703 seconds with return value 16
Press any key to continue . . .
```


Roll No:BID1900

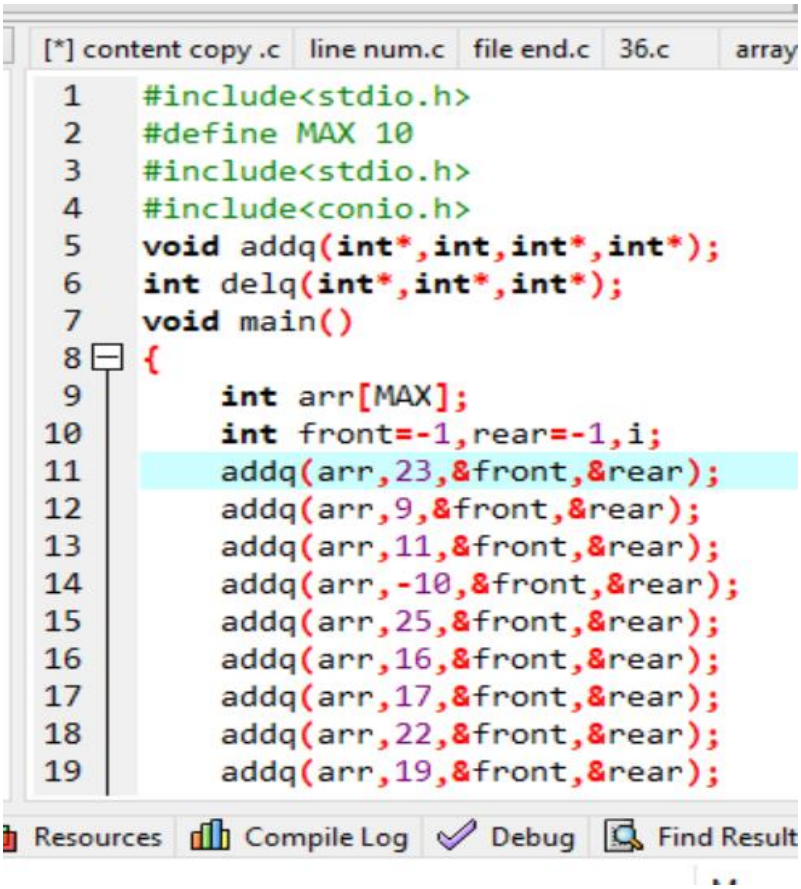
Date:29/9/2020

Exercise 40: Implement the concept of Queue using Data structures

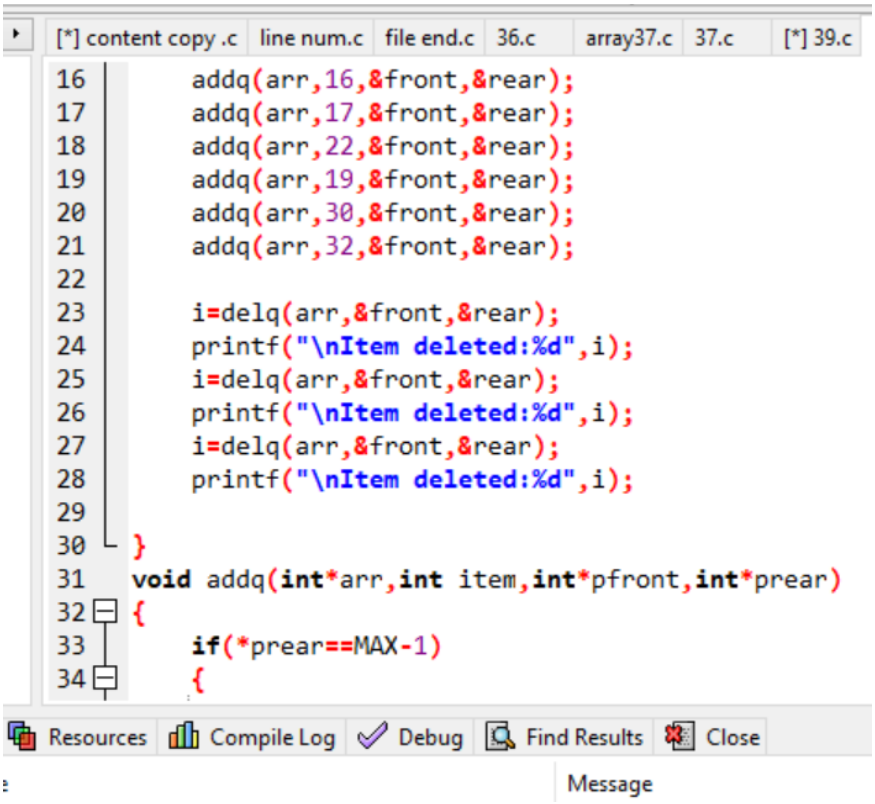
Algorithm:

Step 1: Start
Step 2: Declaring the functions for addq, delq.
Step 3: In main function , declare array of size 10.
Step 4: Call addq function that adds elements to the array
Step 5: Call delq function to delete elements from array
Step 6: Define functions after the main function
Step 7: By conditional operators , define all the functions
Step 8: Print the results
Step 9: Stop

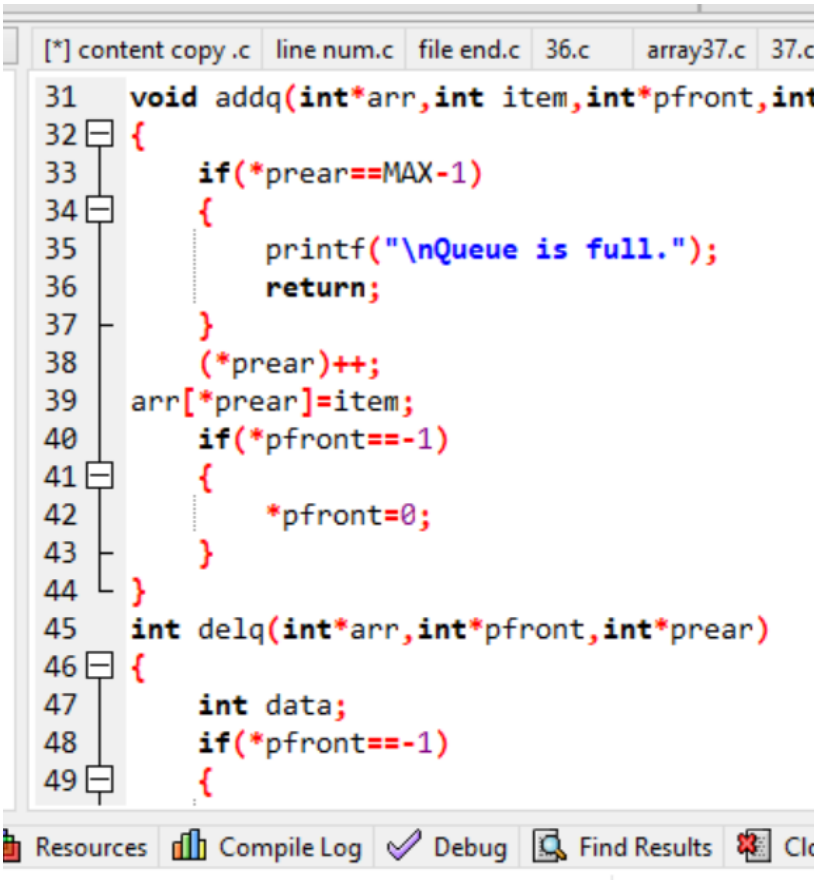
C code:



Roll No:BID1900



```
[*] content copy .c | line num.c | file end.c | 36.c | array37.c | 37.c | [*] 39.c
16      addq(arr,16,&front,&rear);
17      addq(arr,17,&front,&rear);
18      addq(arr,22,&front,&rear);
19      addq(arr,19,&front,&rear);
20      addq(arr,30,&front,&rear);
21      addq(arr,32,&front,&rear);
22
23      i=delq(arr,&front,&rear);
24      printf("\nItem deleted:%d",i);
25      i=delq(arr,&front,&rear);
26      printf("\nItem deleted:%d",i);
27      i=delq(arr,&front,&rear);
28      printf("\nItem deleted:%d",i);
29
30  }
31  void addq(int*arr,int item,int*pfront,int*prear)
32  {
33      if(*prear==MAX-1)
34      {
Resources | Compile Log | Debug | Find Results | Close
Message
```



```
[*] content copy .c | line num.c | file end.c | 36.c | array37.c | 37.c
31  void addq(int*arr,int item,int*pfront,int
32  {
33      if(*prear==MAX-1)
34      {
35          printf("\nQueue is full.");
36          return;
37      }
38      (*prear)++;
39  arr[*prear]=item;
40      if(*pfront==-1)
41      {
42          *pfront=0;
43      }
44  }
45  int delq(int*arr,int*pfront,int*prear)
46  {
47      int data;
48      if(*pfront==-1)
49      {
Resources | Compile Log | Debug | Find Results | Cl
```


Roll No:BID1900

```
{
    int data;
    if(*pfront==-1)
    {
        printf("\nQueue is Empty.");
        return NULL;
    }
    data=arr[*pfront];
    arr[*pfront]=0;
    if(*pfront==*prear)
    {
        *pfront=*prear=-1;
    }
    else
    {
        (*pfront)++;
    }
    return data;
}
```

Output:

```
C:\Users\Prasad\Documents\40.exe

Queue is full.
Item deleted:23
Item deleted:9
Item deleted:11
--------------------------------
Process exited after 0.05662 seconds with return value 16
Press any key to continue . . .
```


Roll No:BID1900

Date:22/9/2020

Virtual Labs Report Writing

Experiment 1

Numerical Representation:

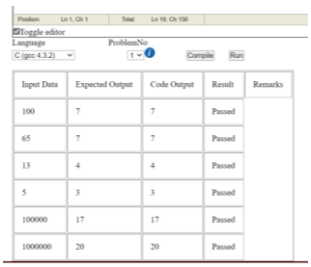
The main aim and the objective of this experiment is to learn to solve problems related to Numerical Representation using Computer Programming. For each problem we have to write a program in C or C++. Each question is evaluted on test cases. An answer is correct only when all the test cases are cleared. The quiz was based on the code and basic logic from the programs. In the experiment we have to write the code for the problem or upload a solution. After that we have to just press the i button get info about the problem where we have to solve.If we are finding it difficult to solve the solution, then we can use the 4 levels of increasingly descriptive hints, but try to use minimum number of hints. As soon as the code is done, compile the code by pressing the compile button. If code is not compling then correct the erroror warnings. Press the execute button to execute the code and see output.

For solving the problems go to the left hand corner of the page you will see the 4th option "experiment" click on it than you will see the option "Go to Lab (Click here)" click on it and you will see a box to solve your question and compile and run the programs.

- Solving Problem 1: Given a positive integer (<= 1000000), find the minimum number of bits required to represent it as a binary number.

```c
#include<stdio.h>
int main()
{
//declare integer i
int i,count=0;
scanf("%d",&i);
//using while loop
while(i)
{
        i>>=1;
        count++;
}
printf("%d\n",count);

return 0;
}
```

- After compiling and running the program :



Input Data	Expected Output	Code Output	Result	Remarks
100	7	7	Passed	
65	7	7	Passed	
13	4	4	Passed	
5	3	3	Passed	
100000	17	17	Passed	
1000000	20	20	Passed	

Roll No:BID1900

Date:22/9/2020

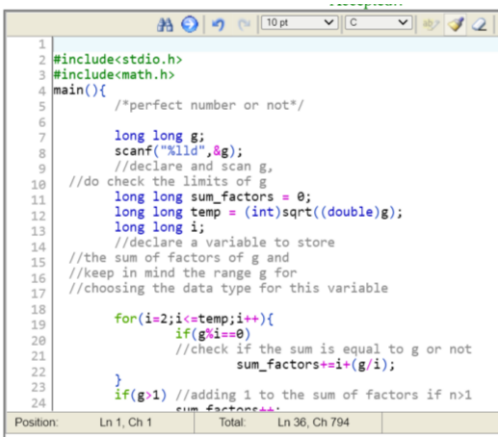
Experiment 2

Beauty of Numbers:

The main aim and the objective of this experiment is to learn to solve problems related to Number Theory using Computer Programming.A number is called perfect, if the sum of all its proper divisors is the number itself. They have also given the input and output specifications input contains a single positive integer(< 10^{10}) , use long long for the number. Interestingly, every positive rational number can be expressed in the form q + 1/E. where q is a non-negative integer and E is either the number 1 or an expression of the same form as above. So such an expression is a continued fraction. For eg., 239/51 is equivalent to 239/51 = 4 + 1/(1+(1/2+(1/(5+1/(2+(1/1)))))). Here we are asked to write a program which takes two integers (eg. 239 and 51) and output a set of integers (eg. 4 1 2 5 2) which are the sequence of q's until E becomes 1.

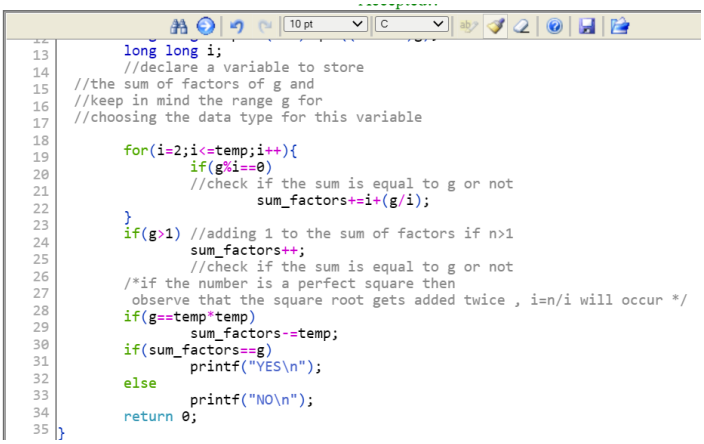
For solving the problems go to the left hand corner of the page you will see the 4th option "experiment" click on it than you will see the option "Go to Lab (Click here)" click on it and you will see a box to solve your question and compile and run the programs.

- Solving Problem 1: Write a program to test, whether a given number is perfect or not. The first perfect number is 6, because 1, 2, and 3 are its proper positive divisors and 1 + 2 + 3 = 6.



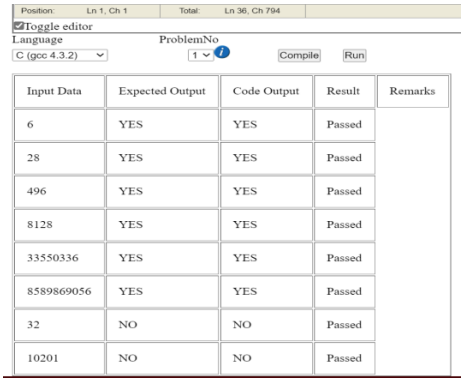
```c
#include<stdio.h>
#include<math.h>
main(){
        /*perfect number or not*/

        long long g;
        scanf("%lld",&g);
        //declare and scan g,
  //do check the limits of g
        long long sum_factors = 0;
        long long temp = (int)sqrt((double)g);
        long long i;
        //declare a variable to store
  //the sum of factors of g and
  //keep in mind the range g for
  //choosing the data type for this variable

        for(i=2;i<=temp;i++){
                if(g%i==0)
                //check if the sum is equal to g or not
                        sum_factors+=i+(g/i);
        }
        if(g>1) //adding 1 to the sum of factors if n>1
```

Position: Ln 1, Ch 1 Total: Ln 36, Ch 794



```c
        long long i;
        //declare a variable to store
  //the sum of factors of g and
  //keep in mind the range g for
  //choosing the data type for this variable

        for(i=2;i<=temp;i++){
                if(g%i==0)
                //check if the sum is equal to g or not
                        sum_factors+=i+(g/i);
        }
        if(g>1) //adding 1 to the sum of factors if n>1
                sum_factors++;
                //check if the sum is equal to g or not
        /*if the number is a perfect square then
         observe that the square root gets added twice , i=n/i will occur */
        if(g==temp*temp)
                sum_factors-=temp;
        if(sum_factors==g)
                printf("YES\n");
        else
                printf("NO\n");
        return 0;
}
```

- After compiling and running the program :



Position: Ln 1, Ch 1 Total: Ln 36, Ch 794

Toggle editor

Language C (gcc 4.3.2) ProblemNo 1 Compile Run

Input Data	Expected Output	Code Output	Result	Remarks
6	YES	YES	Passed	
28	YES	YES	Passed	
496	YES	YES	Passed	
8128	YES	YES	Passed	
33550336	YES	YES	Passed	
8589869056	YES	YES	Passed	
32	NO	NO	Passed	
10201	NO	NO	Passed	

Roll No:BID1900

Date:22/9/2020

Experiment 3

More on Numbers:

The ability to understand what the end goal of the problem is, and what rules could be applied represents the key to solving the problem. Sometimes the problem requires abstract thinking or coming up with a creative solution. The main aim and the objective of this experiment to learn to solve problems related to Number Theory using Computer

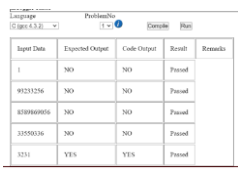
ProgrammingThe Input and output Specifications are also given :input contains a single positive integer(< 10^{10}). Use long long for the number and for the output "YES" or "NO" (quotes for clarity) corresponding to whether the number in input is divisible by 9 or not. In the second problem a whole number is said to be CIRCULAR if, when you multiply the number by its units decimal digit, the result is the number with its decimal digits rotated to the right, where the units digit becoming its high-order digit.

For solving the problems go to the left hand corner of the page you will see the 4th option "experiment" click on it than you will see the option "Go to Lab (Click here)" click on it and you will see a box to solve your question and compile and run the programs.

- Solving Problem 1: In this experiment the problem number 1 says that given a positive integer there is an easy way to check whether this integer is divisible by 9 or not. This was very popular before mod operator was invented. The solution is simple, if we add the digits of the number repeatedly and finally get 9, then the given number is divisible by 9, else it is not divisible by 9.

```c
#include<stdio.h>
//declare and scan g
int sum_digits(long long g){
        return g?(g%10)+sum_digits(g/10):0;
}


int main()
{
        long long g;
        scanf("%lld",&g);
        //we go on calculating the sum of the digits
          //of the result;ing number until it becomes less than or equal to 9
         //while g is not a single digit number g= sum_digits(g)
        while(g>9)
          g=sum_digits(g);
        if(g==9)
                printf("YES\n");
        else
                printf("NO\n");
        return 0;
}
```

- After compiling and running the program :



Input Data	Expected Output	Code Output	Result	Remarks
1	NO	NO	Passed	
93253256	NO	NO	Passed	
8589869056	NO	NO	Passed	
33550336	NO	NO	Passed	
3231	YES	YES	Passed	

Date:22/9/2020

Roll No:BID1900

Experiment 4

Factorials:

The main aim and the objective of this experiment is to give an opportunity for learning and better understanding of using computer programming as a tool to solve basic to advanced problems.Although the factorial function has its roots in combinatorics, formulas involving factorials occur in many areas of mathematics. There aren! different ways of arranging n distinct objects into a sequence, the permutations of those objects. Factorials occur in algebra for various reasons, such as coefficients of the binomial formula, or through averaging over permutations for symmetrization of certain operations. Factorials also turn up in calculus; for example they occur in the denominators of the terms of Taylor's formula, basically to compensate for the fact that the n-th derivative of xn is n!. Factorials are also used extensively in probability theory. Factorials can be useful to facilitate expression manipulation.

For solving the problems go to the left hand corner of the page you will see the 4th option "experiment" click on it than you will see the option "Go to Lab (Click here)" click on it and you will see a box to solve your question and compile and run the programs.

- Solving Problem 1:Given a positive integer, find the number of digits in the factorial of the number.

```
#include <stdio.h>
#include <math.h>
main(){
int g,i;
double result = 0;
scanf("%d",&g);

for(i=1;i<=g;i++)
        result+=log10(i);

printf("%d\n",(int)result+1);
return 0;
}
```

- After compiling and running the program :

Language: C (gcc 4.3.2) ProblemNo: 1 Compile Run

Input Data	Expected Output	Code Output	Result	Remarks
100	158	158	Passed	
65	91	91	Passed	
13	10	10	Passed	
5	3	3	Passed	
100000	456574	456574	Passed	
1000000	5565709	5565709	Passed	

Date:22/9/2020

Roll No:BID1900

Experiment 5

String Operations:

Problem solving consists of using generic or ad hoc methods in an orderly manner to find solutions to problems. Some of the problem-solving techniques developed and used in philosophy, artificial,intelligence, computer,science, engineering, mathematics,

or medicine are related to mental problem-solving techniques studied in psychology.A palindrome is a string which reads same wheter it is read from start to end of from end to the start. In literature, however, even the capitalization of words and punctuations are ignored. So, "Rats live on no evil stars." is also a palindrome. Detection of such palindromes can be done by creating a new string in which all the punctuations have been removed and all the upper case characters have been converted to lower case characters, and then doing a test for plaindome on the new string.

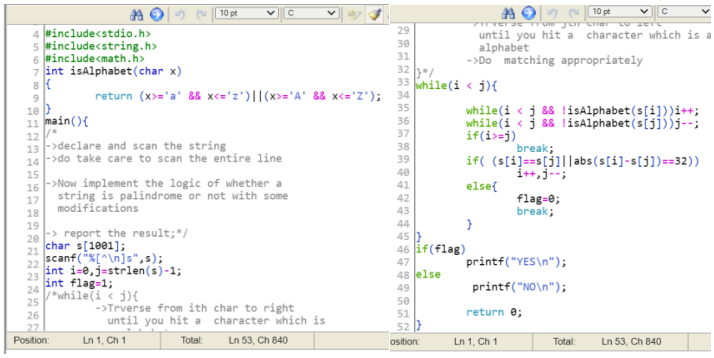
Alternatively, one can create two indices and put them at the end and beginning of the string. If a character at any of the indices is not an alphabet the corresponding indice is moved until an alphabet is encountered. If the alphabet is in Upper case, then it is converted into lower

case. Now, the values at the indices are compared and if they are not same then the string is not a palindome. Otherwise, the check is repeated for new values of indices obtained by moving them by suffiecient amount so that an alphabet is encountered. If at any time the indices are coincident or the indices cross each other, then the string is a palindrome. Inorder to obtain string2 from string1 by deleting 0 or more characters it is fairly obvious that the characters in string2 must appear in the same order as in string1. String Operations

is to learn to solve problems related to Strings using Computer Programming.

For solving the problems go to the left hand corner of the page you will see the 4th option “experiment” click on it than you will see the option “Go to Lab (Click here)” click on it and you will see a box to solve your question and compile and run the programs.

- Solving Problem 1: Write a program that detects palindromes. Your program should accept a string from the key board and output YES/NO.



```
#include<stdio.h>
#include<string.h>
#include<math.h>
int isAlphabet(char x)
{
        return (x>='a' && x<='z')||(x>='A' && x<='Z');
}
main(){
/*
->declare and scan the string
->do take care to scan the entire line

->Now implement the logic of whether a
  string is palindrome or not with some
  modifications

-> report the result;*/
char s[1001];
scanf("%[^\n]s",s);
int i=0,j=strlen(s)-1;
int flag=1;
/*while(i < j){
        ->Trverse from ith char to right
          until you hit a  character which is
          alphabet
        ->Do  matching appropriately
}*/
while(i < j){

        while(i < j && !isAlphabet(s[i]))i++;
        while(i < j && !isAlphabet(s[j]))j--;
        if(i>=j)
                break;
        if( (s[i]==s[j]||abs(s[i]-s[j])==32))
                i++,j--;
        else{
                flag=0;
                break;
        }
}
if(flag)
        printf("YES\n");
else
         printf("NO\n");

        return 0;
}
```

- After compiling and running the program 

Roll No:BID1900

Date:22/9/2020

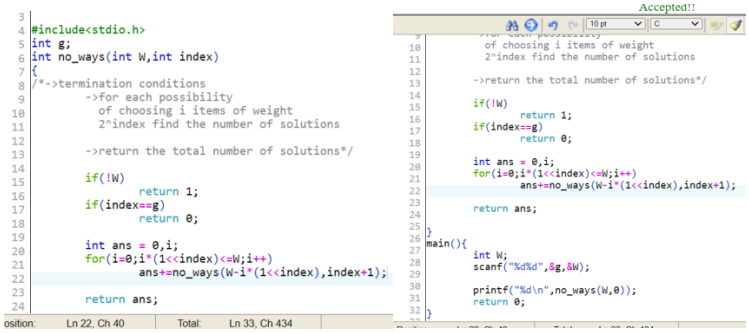
Experiment 6

Recursion:

The process in which a function calls itself directly or indirectly is called recursion and the corresponding function is called as recursive function. Using recursive algorithm, certain problems can be solved quite easily. Examples of such problems are Towers of Hanoi (TOH), Inorder/Preorder/Postorder Tree Traversals, DFS of Graph, etc.Recursion means the repeated application of a recursive procedure or definition.one of the most easiest things to code but really hard to debug. Recursion is a very powerful tool to solve problems. All loops(while, for , do while) can be simulated using recursion. Also the main step of the dynamic programming is recursion which is useful in solving many algorithmically difficult problems. Soin the first problem we are given scales for weighing loads. On the left side lies a single stone of known weight W < 2N.

For solving the problems go to the left hand corner of the page you will see the 4th option "experiment" click on it than you will see the option "Go to Lab (Click here)" click on it and you will see a box to solve your question and compile and run the programs.

- Solving Problem 1: We own a set of N different weights, weighing 1, 2, 4, ..., 2^N-1 units of mass respectively. Determine how many possible ways there are of placing some weights on the sides of the scales, so as to balance them (we have to put them in a state of equilibrium).

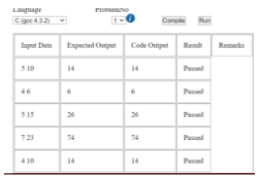


```
#include<stdio.h>
int g;
int no_ways(int W,int index)
{
/*->termination conditions
        ->for each possibility
          of choosing i items of weight
          2^index find the number of solutions

        ->return the total number of solutions*/

        if(!W)
                return 1;
        if(index==g)
                return 0;

        int ans = 0,i;
        for(i=0;i*(1<<index)<=W;i++)
                ans+=no_ways(W-i*(1<<index),index+1);

        return ans;
```

Accepted!!

```
          of choosing i items of weight
          2^index find the number of solutions

        ->return the total number of solutions*/

        if(!W)
                return 1;
        if(index==g)
                return 0;

        int ans = 0,i;
        for(i=0;i*(1<<index)<=W;i++)
                ans+=no_ways(W-i*(1<<index),index+1);

        return ans;
}
main(){
        int W;
        scanf("%d%d",&g,&W);

        printf("%d\n",no_ways(W,0));
        return 0;
}
```

- After compiling and running the program



Input Data	Expected Output	Code Output	Result	Remarks
5 10	14	14	Passed	
4 6	6	6	Passed	
5 15	26	26	Passed	
7 23	74	74	Passed	
4 10	14	14	Passed	

Roll No:BID1900

Date:22/9/2020

Experiment 7

Advanced Arithmatic:

The main aim of this experiment is to learn to solve problems related to Advanced Arithmatic using Computer Programming. Problem solving is used when products or processes fail, so corrective action can be taken to prevent further failures. It can also be applied to a product or process prior to an actual failure event when a potential problem can be predicted and analyzed, and mitigation applied so the problem never occurs. Techniques such as failure mode and effects analysis can be used to proactively reduce the likelihood of problems occurring. Here for the first problem if numbers are to big to fit into standard data types, then we can use strings to store the number. So in this case, addition of two number would require doing digit by digit addition along with required carry over getting transferred to the next higher order cell.

For solving the problems go to the left hand corner of the page you will see the 4th option "experiment" click on it than you will see the option "Go to Lab (Click here)" click on it and you will see a box to solve your question and compile and run the programs.

- Solving Problem 1: The task is to accomplish addition of two large positive numbers

```c
#include<stdio.h>
#include<string.h>
main(){
        char s1[51],s2[51];
        scanf("%s %s",s1,s2);
        char sum[52];
        /*for i= strlen(str1)-1,j=strlen(str2)-1; i>=0||j>=0 ;i--,j--

        ->value1 = (i>=0)?str1[i]-'0':0
        ->value2 = (j>=0)?str2[j]-'0':0
        ->output[k++] = (value1+value2+carry)%10;
        ->carry = (value1+value2+carry)/10; */

        int l1 = strlen(s1),l2 = strlen(s2),index = 0,carry=0,value,i,j;
        for(i=l1-1,j=l2-1;i>=0 || j>=0;){

                value = carry;
                if(i>=0){
                        value+=(s1[i]-'0');
                        i--;
                }
                if(j>=0){
```

```c
        int l1 = strlen(s1),l2 = strlen(s2),index = 0,carry=0,value,i,j;
        for(i=l1-1,j=l2-1;i>=0 || j>=0;){

                value = carry;
                if(i>=0){
                        value+=(s1[i]-'0');
                        i--;
                }
                if(j>=0){
                        value+=(s2[j]-'0');
                        j--;
                }
                sum[index++]=(char)(value%10 + '0');
                carry = value/10;
        }
        //take care of the cases like 99+1
        if(carry)
                sum[index++]='1';

        for(i=index-1;i>=0;i--)
                printf("%c",sum[i]);
        puts("");
        return 0;
}
```

- After compiling and running the program

C (gcc 4.3.2) | 1 | Compile | Run

Input Data	Expected Output	Code Output	Result	Remarks
100 58	158	158	Passed	
3232893293293928933283928392839283928932329	3283928716128613258325222	3283928716128613258325222	Passed	
323289329329392893999999392839283908932329	10000002625732577194252222	10000002625732577194252222	Passed	
23 79	102	102	Passed	
3283928392839283928932329 323289329329392893	3283928716128613258325222	3283928716128613258325222	Passed	

Roll No:BID1900

Date:22/9/2020

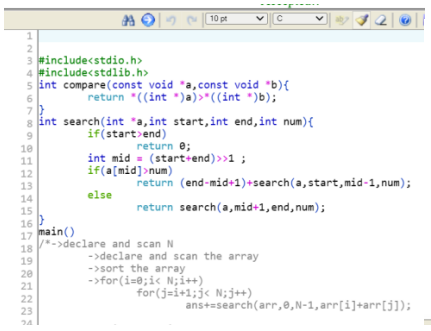
Experiment 8

Searching and Sorting:

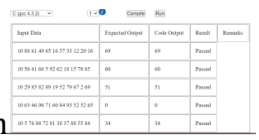
Searching and Sorting are very important areas in computer science. Efficient sorting algorithms are required for optimizing the use of other algorithms that require sorted lists to work correctly, for example to do binary search sorting is required. Two problemsare given based on variant of binary search and sorting.Searching Algorithms are designed to check for an element or retrieve an element from any data structure where it is stored. Based on the type of search operation, these algorithms are generally classified into two categories:Sequential Search: In this, the list or array is traversed sequentially and every element is checked. For example: Linear Search.Interval Search: These algorithms are specifically designed for searching in sorted data-structures. These type of searching algorithms are much more efficient than Linear Search as they repeatedly target the center of the search structure and divide the search space in half. For Example: Binary Search.For each problem we have to write a program in C or C++. Each question is evaluted on test cases. An answer is correct only when all the test cases are cleared.

- Solving Problem 1: You are given some set of numbers. By choosing three numbers we may be able to construct a triangle. In this problem we ask you to find out the number of invalid triangles formed from these set of numbers. The degenerate traiangles are considered to be valid traiangles.
- Algorithm:-

  1.declare and scan N
  2.declare and scan the array
  3.sort the array
  4.for each pair (i,j) find
  5.out how many k's exist
  6.such that arr[i]+arr[j]<arr[k]
  7.print the final answer



```c
#include<stdio.h>
#include<stdlib.h>
int compare(const void *a,const void *b){
        return *((int *)a)>*((int *)b);
}
int search(int *a,int start,int end,int num){
        if(start>end)
                return 0;
        int mid = (start+end)>>1 ;
        if(a[mid]>num)
                return (end-mid+1)+search(a,start,mid-1,num);
        else
                return search(a,mid+1,end,num);
}
main()
/*->declare and scan N
        ->declare and scan the array
        ->sort the array
        ->for(i=0;i< N;i++)
                for(j=i+1;j< N;j++)
                        ans+=search(arr,0,N-1,arr[i]+arr[j]);
        ->print the final answer
}*/
{
        int N;
        scanf("%d",&N);
        int arr[N];
        int i,j;
        for(i=0;i< N;i++)
                scanf("%d",arr+i);
        qsort(arr,N,sizeof(int),compare);
        int ans=0;
        for(i=0;i< N;i++)
                for(j=i+1;j< N;j++)
                        ans+=search(arr,0,N-1,arr[i]+arr[j]);
        printf("%d\n",ans);
        return 0;
}
```



- After compiling and running the program

Roll No:BID1900

Date:22/9/2020

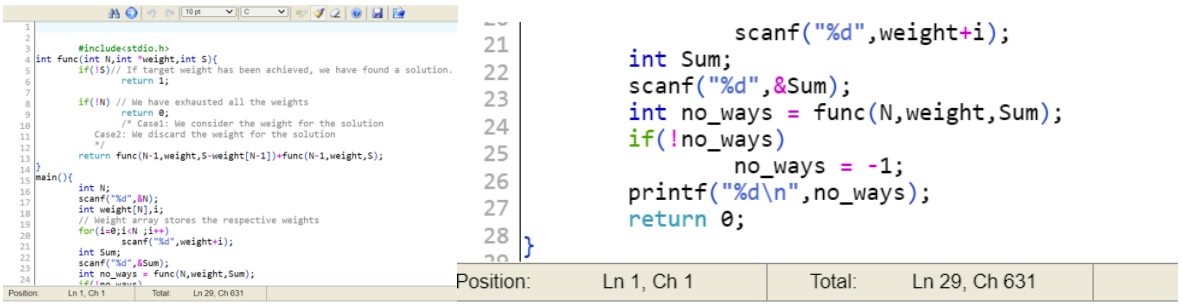
Experiment 9

Permutations:

Permutations are used in almost every branch of mathematics, and in many other fields of science. In computer science, they are used for analyzing sorting algorithms; in quantum physics, for describing states of particles; and in biology, for describing RNA sequences.
The number of permutations of n distinct objects is n factorial, usually written as n!, which means the product of all positive integers less than or equal to n.Permutations occur, in more or less prominent ways, in almost every domain of mathematics. They often arise when different orderings on certain finite sets are considered, possibly only because one wants to ignore such orderings and needs to know how many configurations are thus identified. For similar reasons permutations arise in the study of sorting algorithms in computer science.

For solving the problems go to the left hand corner of the page you will see the 4th option "experiment" click on it than you will see the option "Go to Lab (Click here)" click on it and you will see a box to solve your question and compile and run the programs.

- Solving Problem 1: Given n distict and unique integral weights and an integral value, Sum. You have to tell in how many ways you can balance that Sum using one or all weights. If there is no combination possible, print -1.

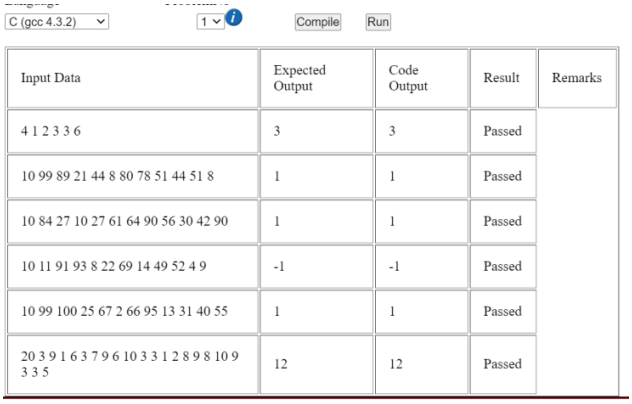


```
#include<stdio.h>
int func(int N,int *weight,int S){
    if(!S)// If target weight has been achieved, we have found a solution.
        return 1;

    if(!N) // We have exhausted all the weights
        return 0;
        /* Case1: We consider the weight for the solution
    Case2: We discard the weight for the solution
     */
    return func(N-1,weight,S-weight[N-1])+func(N-1,weight,S);
}
main(){
    int N;
    scanf("%d",&N);
    int weight[N],i;
    // Weight array stores the respective weights
    for(i=0;i<N ;i++)
        scanf("%d",weight+i);
    int Sum;
    scanf("%d",&Sum);
    int no_ways = func(N,weight,Sum);
```

```
        scanf("%d",weight+i);
int Sum;
scanf("%d",&Sum);
int no_ways = func(N,weight,Sum);
if(!no_ways)
        no_ways = -1;
printf("%d\n",no_ways);
return 0;
}
```

Position: Ln 1, Ch 1 Total: Ln 29, Ch 631

- After compiling and running the program



C (gcc 4.3.2) 1 Compile Run

Input Data	Expected Output	Code Output	Result	Remarks
4 1 2 3 3 6	3	3	Passed	
10 99 89 21 44 8 80 78 51 44 51 8	1	1	Passed	
10 84 27 10 27 61 64 90 56 30 42 90	1	1	Passed	
10 11 91 93 8 22 69 14 49 52 4 9	-1	-1	Passed	
10 99 100 25 67 2 66 95 13 31 40 55	1	1	Passed	
20 3 9 1 6 3 7 9 6 10 3 3 1 2 8 9 8 10 9 3 3 5	12	12	Passed	

Roll No:BID1900

Date:22/9/2020

Experiment 10

Sequences:

Thisexperiment is nothing but just aintroduction to two very famous algorithmic problems.One problem is longest continuous decreasing subsequence problem and other one is longest increasing subsequence problem.. In the experiment we have to write the code for the problem or upload a solution. After that we have to just press the i button get info about the problem where we have to solve.If we are finding it difficult to solve the solution, then we can use the 4 levels of increasingly descriptive hints, but try to use minimum number of hints. As soon as the code is done, compile the code by pressing the compile button. If code is not compling then correct the erroror warnings. Press the execute button to execute the code and see output.

For solving the problems go to the left hand corner of the page you will see the 4th option "experiment" click on it than you will see the option "Go to Lab (Click here)" click on it and you will see a box to solve your question and compile and run the programs.

- Solving Problem 1:Given a list of n distinct numbers (where n <= 1000), find the length of the longest monotone decreasing subsequence.
- Algorithm:

  1.declare N and scan N

  2.declare an array of size of N and scan it

  3.declare variables to store themaximum size of the continuoussubsequence and the size of current decreasing subsequence

  4.Iterate over the array and update the variables

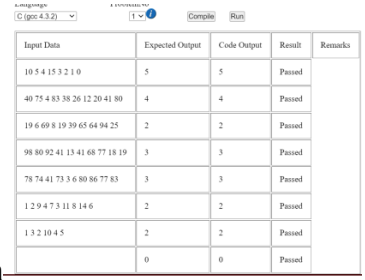
  5.print the maximum size of decreasing subsequence.

Accepted!!

```c

#include<stdio.h>
#include<stdlib.h>
int main(){
int num;
        int length=1;
        int max_length=0,i,curr_num;
        if(fscanf(stdin,"%d",&num)!=EOF) /* If there is no number in the input file
maximum length must be 0,
                                otherwise it is atleast 1*/
                max_length=1;

        while(fscanf(stdin,"%d",&curr_num)!=EOF){
    // Scan and compute longest subsequence till EOF
                if(curr_num<num)
                        length++;
                else
                        length = 1;
                if(max_length<length)
                        max_length = length;
                num = curr_num;
        }
        printf("%d\n",max_length);
```


Roll No:BID1900

❖ After compiling and running the program



C (gcc 4.3.2) 1 Compile Run

Input Data	Expected Output	Code Output	Result	Remarks
10 5 4 15 3 2 1 0	5	5	Passed	
40 75 4 83 38 26 12 20 41 80	4	4	Passed	
19 6 69 8 19 39 65 64 94 25	2	2	Passed	
98 80 92 41 13 41 68 77 18 19	3	3	Passed	
78 74 41 73 3 6 80 86 77 83	3	3	Passed	
1 2 9 4 7 3 11 8 14 6	2	2	Passed	
1 3 2 10 4 5	2	2	Passed	
	0	0	Passed	

Roll No:BID1900

Roll No:BID1900

Roll No:BID1900