

INDEX

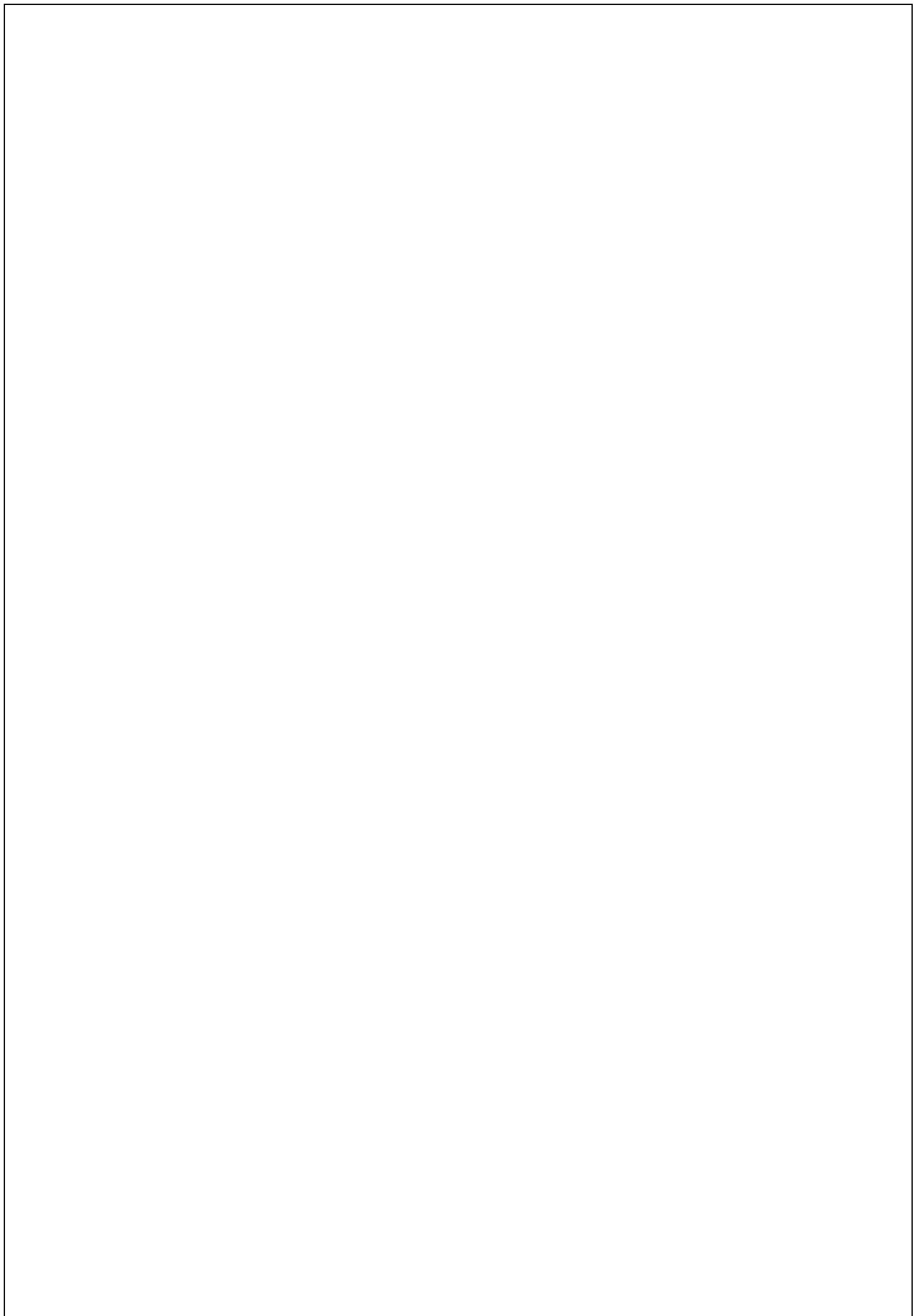
Sl. No.	Exercise title	Date of performance	Date of correction	Page number	Signature (Internal Supervisor)
1	Perform program for swapping of two numbers without using a temporary variable	9/9/2020		1	
2	Perform program for swapping of two numbers using a temporary variable	9/9/2020		2	
3	Calculate sum 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:



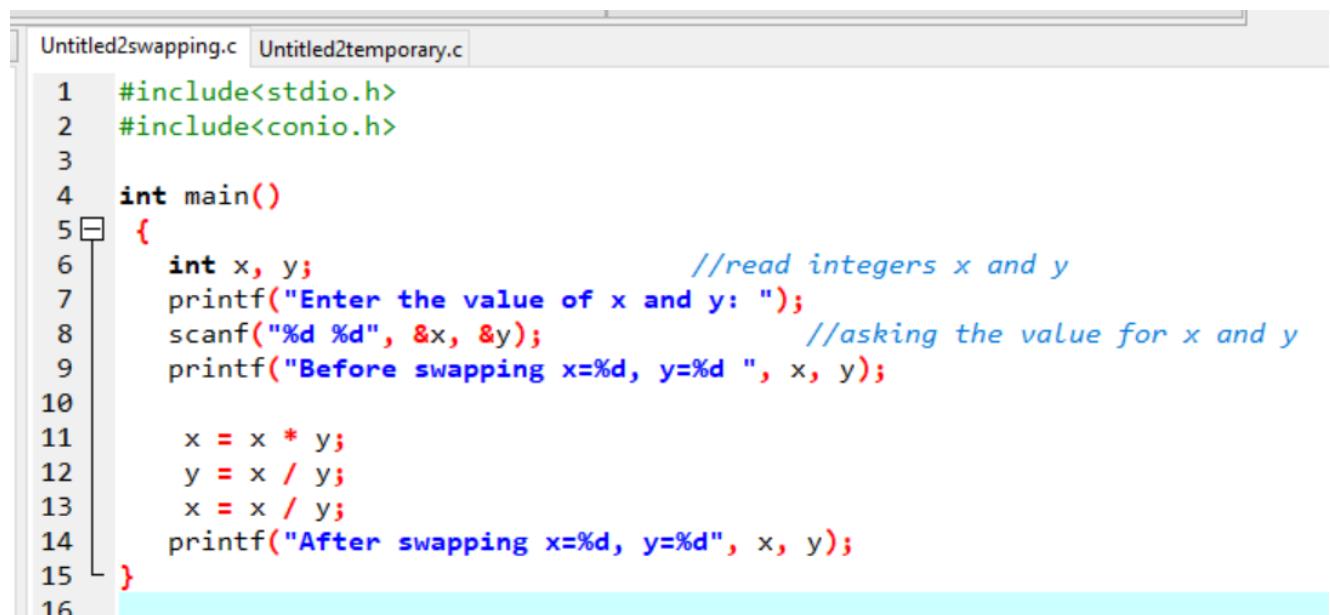
Date:9/9/2020

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

Algorithm:

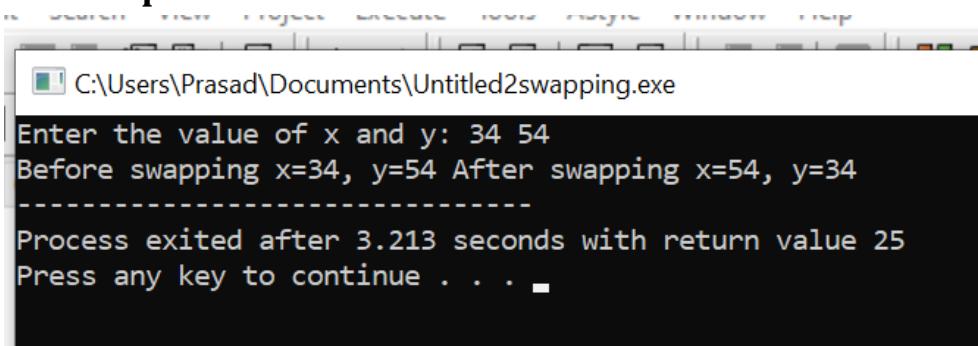
- Step 1: Start
- Step 2: Read x and y
- Step 3: $x = x * y;$
- Step 4: $y = x / y;$
- Step 5: $x = x / y;$
- Step 6: Print x and y
- Step 7: Stop

C code:



```
Untitled2swapping.c Untitled2temporary.c
1 #include<stdio.h>
2 #include<conio.h>
3
4 int main()
5 {
6     int x, y; //read integers x and y
7     printf("Enter the value of x and y: ");
8     scanf("%d %d", &x, &y); //asking the value for x and y
9     printf("Before swapping x=%d, y=%d ", x, y);
10
11    x = x * y;
12    y = x / y;
13    x = x / y;
14    printf("After swapping x=%d, y=%d", x, y);
15 }
16
```

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 . . .
```

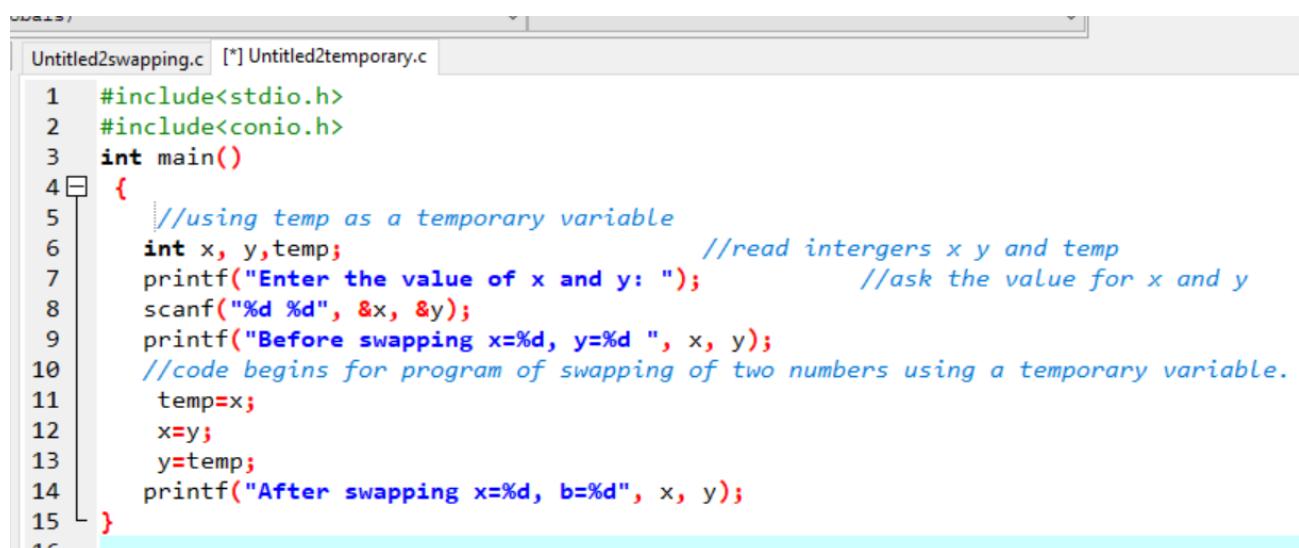
Date:9/9/2020

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

Algorithm:

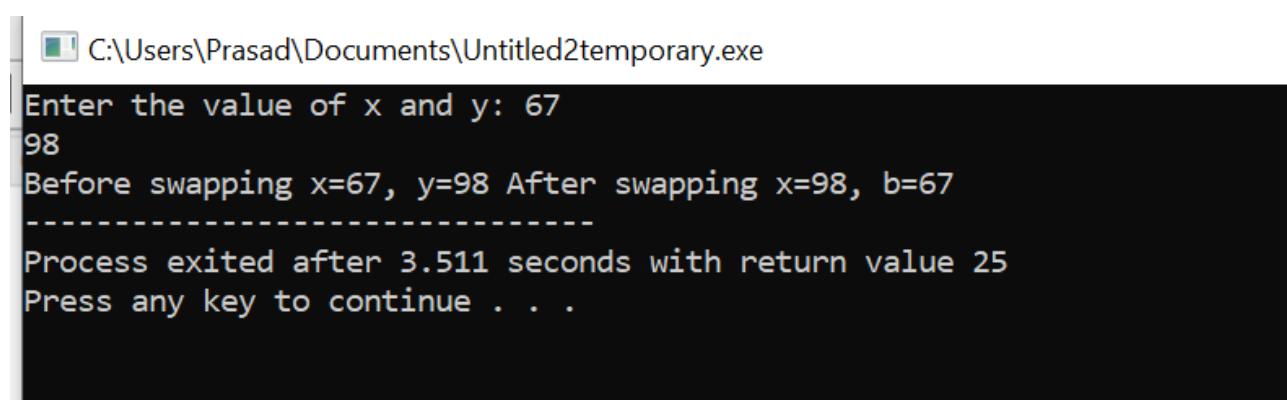
- Step 1 : Start
- Step 2 : Declare integer x,y,temp
- Step 3 : READ x, y
- Step 4 : temp = x
- Step 5 :x= y
- Step 6 :y = temp
- Step 7 : PRINT x, y
- Step 8 : Stop

C code:



```
Untitled2swapping.c [*] Untitled2temporary.c
1 #include<stdio.h>
2 #include<conio.h>
3 int main()
4 {
5     //using temp as a temporary variable
6     int x, y,temp;           //read intergers x y and temp
7     printf("Enter the value of x and y: ");      //ask the value for x and y
8     scanf("%d %d", &x, &y);
9     printf("Before swapping x=%d, y=%d ", x, y);
10    //code begins for program of swapping of two numbers using a temporary variable.
11    temp=x;
12    x=y;
13    y=temp;
14    printf("After swapping x=%d, b=%d", x, y);
15 }
```

Output:



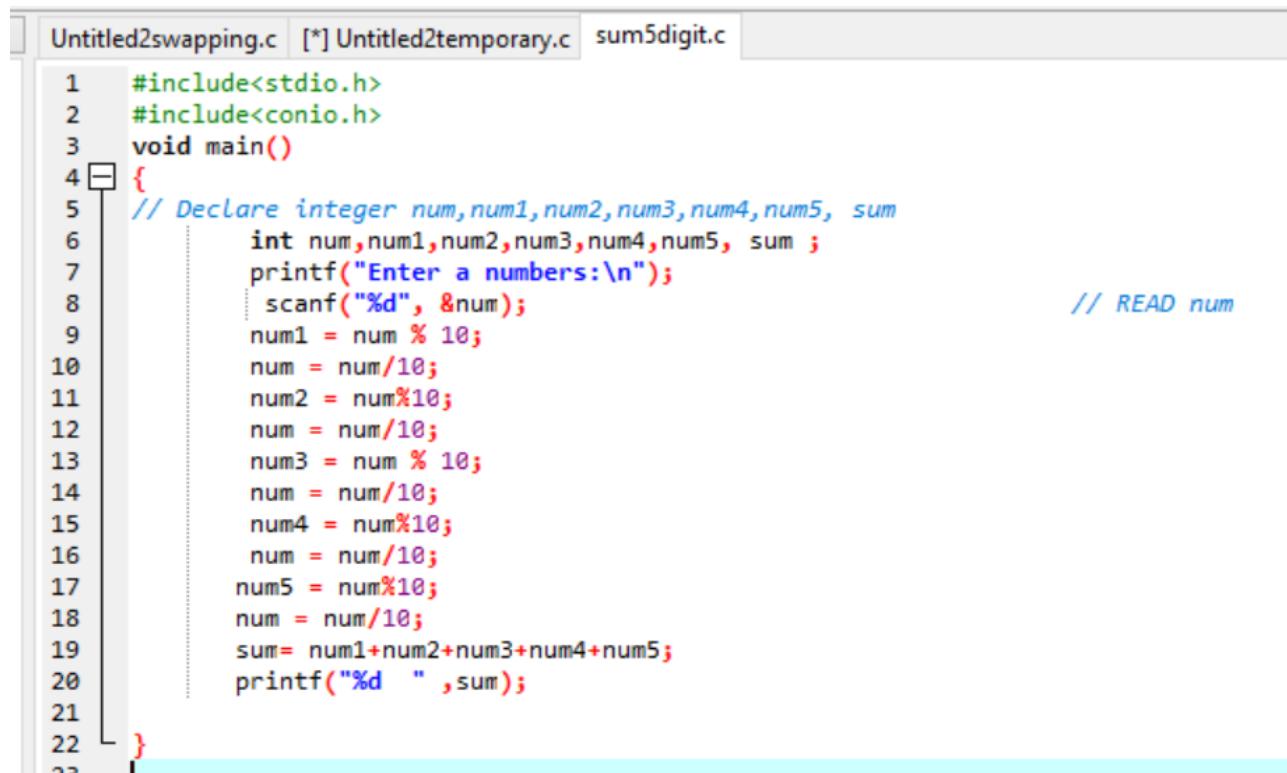
```
C:\Users\Prasad\Documents\Untitled2temporary.exe
Enter the value of x and y: 67
98
Before swapping x=67, y=98 After swapping x=98, b=67
-----
Process exited after 3.511 seconds with return value 25
Press any key to continue . . .
```

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:

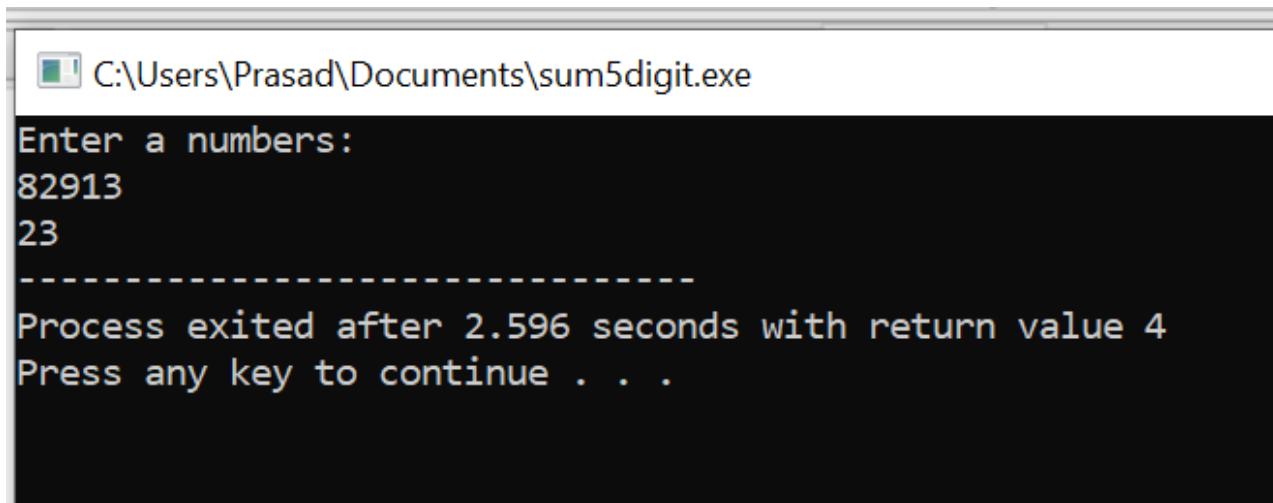


```

1 #include<stdio.h>
2 #include<conio.h>
3 void main()
4 {
5   // Declare integer num,num1,num2,num3,num4,num5, sum
6   int num,num1,num2,num3,num4,num5, sum ;
7   printf("Enter a numbers:\n");
8   scanf("%d", &num);                                // READ num
9   num1 = num % 10;
10  num = num/10;
11  num2 = num%10;
12  num = num/10;
13  num3 = num % 10;
14  num = num/10;
15  num4 = num%10;
16  num = num/10;
17  num5 = num%10;
18  num = num/10;
19  sum= num1+num2+num3+num4+num5;
20  printf("%d ",sum);
21
22 }

```

Output:



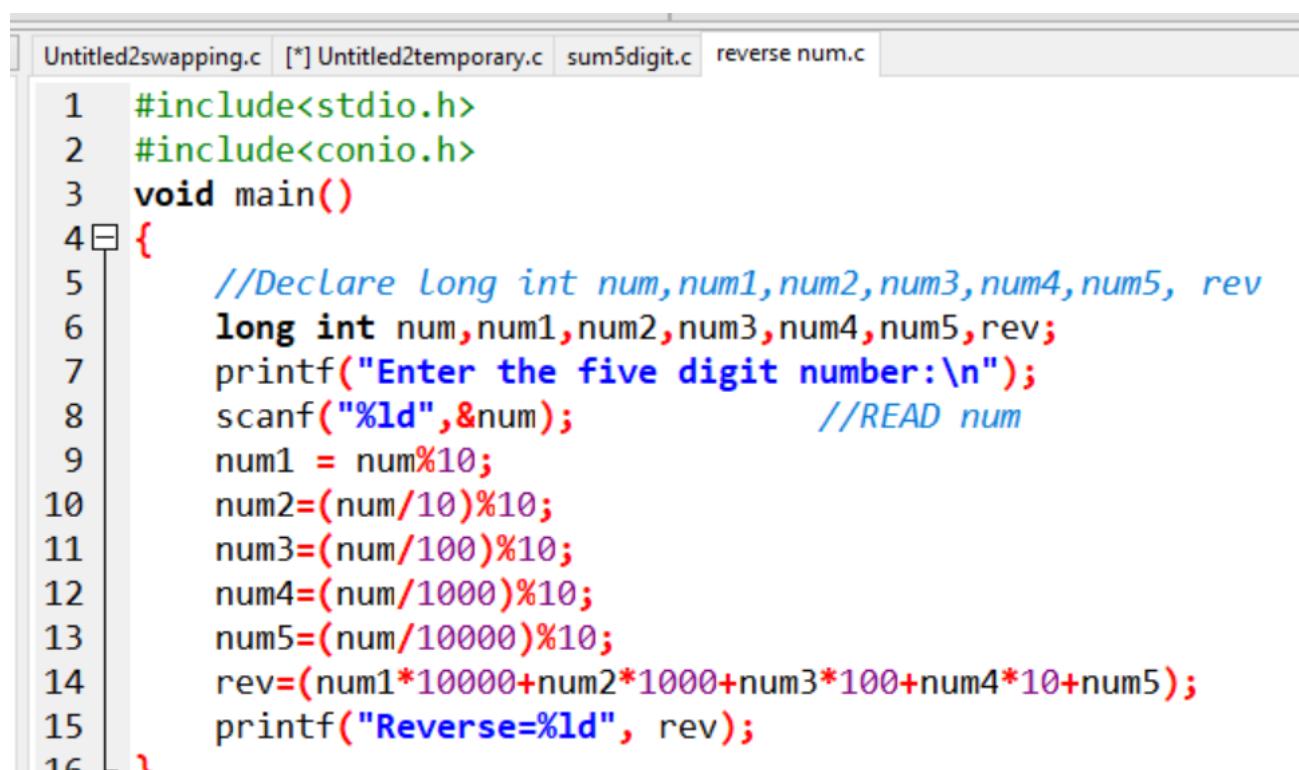
```
C:\Users\Prasad\Documents\sum5digit.exe
Enter a numbers:
82913
23
-----
Process exited after 2.596 seconds with return value 4
Press any key to continue . . .
```

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:



The screenshot shows a code editor window with multiple tabs at the top: Untitled2swapping.c, Untitled2temporary.c, sum5digit.c, and reverse num.c. The reverse num.c tab is active. The code itself is as follows:

```
1 #include<stdio.h>
2 #include<conio.h>
3 void main()
4 {
5     //Declare Long int num,num1,num2,num3,num4,num5, rev
6     long int num,num1,num2,num3,num4,num5,rev;
7     printf("Enter the five digit number:\n");
8     scanf("%ld",&num);           //READ num
9     num1 = num%10;
10    num2=(num/10)%10;
11    num3=(num/100)%10;
12    num4=(num/1000)%10;
13    num5=(num/10000)%10;
14    rev=(num1*10000+num2*1000+num3*100+num4*10+num5);
15    printf("Reverse=%ld", rev);
16 }
```

Output:

```
C:\Users\Prasad\Documents\reverse num.exe
Enter a five digit number:53627
Reverse number:72635
-----
Process exited after 3.129 seconds with return value 21
Press any key to continue . . .
```

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)

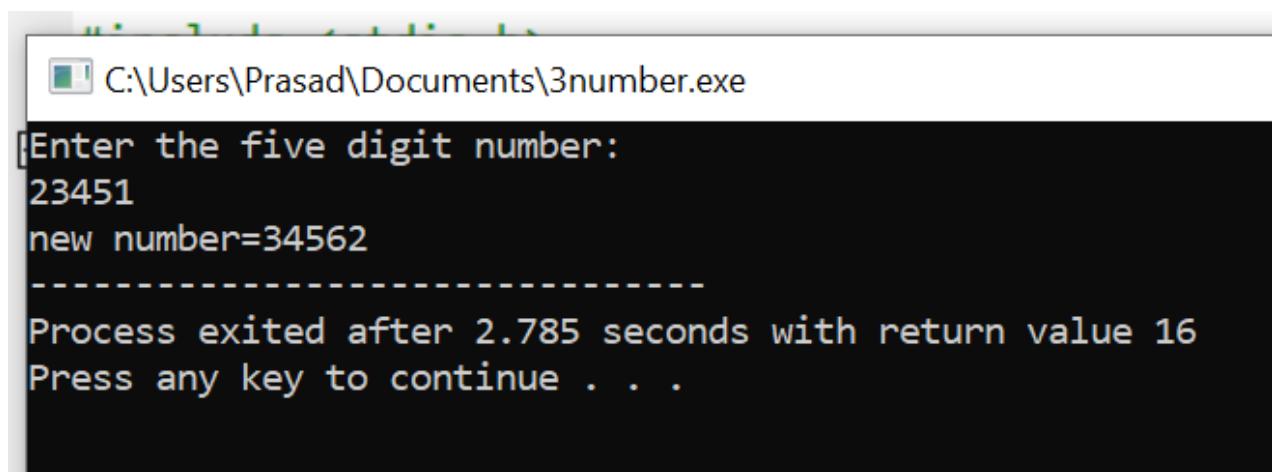
Step 10 : PRINT newnum

Step 11 : Stop

C code:

```
Untitled2swapping.c [*] Untitled2temporary.c sum5digit.c reverse num.c [*] 3number.c
1 #include<conio.h>
2 #include <stdio.h>
3 void main()
4 {
5     //Declare long int num,num1,num2,num3,num4,num5, newnum
6     long int num,num1,num2,num3,num4,num5,newnum;
7     printf("Enter the five digit number:\n");
8     scanf("%ld",&num); //READ num
9     num1 =(num%10)+1;
10    num2=((num/10)%10)+1;
11    num3=((num/100)%10)+1;
12    num4=((num/1000)%10)+1;
13    num5=((num/10000)%10)+1;
14    newnum=(num5*10000+num4*1000+num3*100+num2*10+num1);
15    printf("new number=%ld", newnum); //PRINT newnum
16 }
17
```

Output:



```
C:\Users\Prasad\Documents\3number.exe
Enter the five digit number:
23451
new number=34562
-----
Process exited after 2.785 seconds with return value 16
Press any key to continue . . .
```

Exercise 6: Find absolute value of a number

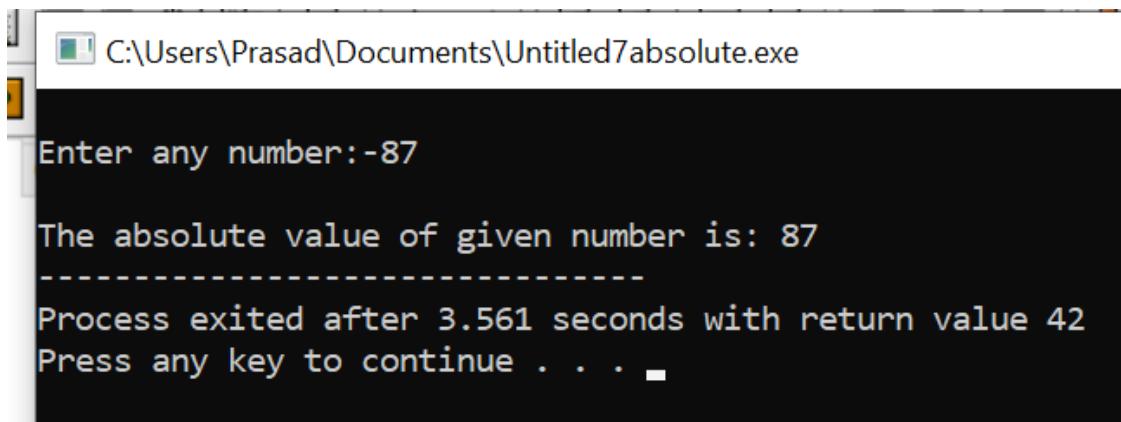
Algorithm:

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
1 #include<stdio.h>
2 #include<conio.h>
3 void main()
4 {
5     int num;          //Declare int num , READ num
6
7
8     printf("\nEnter any number:");
9
10    scanf("%d",&num);      //reading entered value from user
11
12    if(num< 0)
13        num = num*-1;
14    printf("\nThe absolute value of given number is: %d", num);
15
16 }
17
```

Output:



```
C:\Users\Prasad\Documents\Untitled7absolute.exe
Enter any number:-87
The absolute value of given number is: 87
-----
Process exited after 3.561 seconds with return value 42
Press any key to continue . . .
```

Date:10/9/2020

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

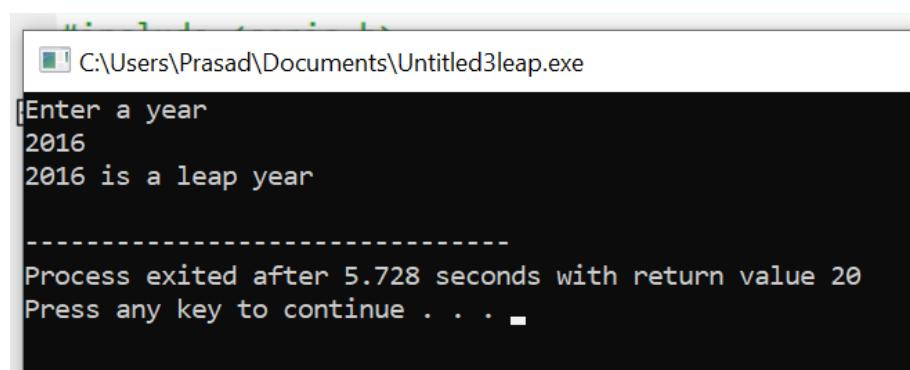
Algorithm:

- 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
1 #include <stdio.h>
2 #include <conio.h>
3 int main()
4 {
5     int year;           //Declare int year
6
7     printf("Enter a year\n");
8     scanf("%d", &year);      //READ year
9
10    if( (year%100 == 0 && year%400 == 0) || (year%100 != 0 && year%4 == 0) )
11    {
12        printf("%d is a leap year\n", year);
13    }
14    else
15    {
16        printf("%d is not a leap year\n", year);
17    }
}
```

Output



```
C:\Users\Prasad\Documents\Untitled3leap.exe
Enter a year
2016
2016 is a leap year
-----
Process exited after 5.728 seconds with return value 20
Press any key to continue . . .
```

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

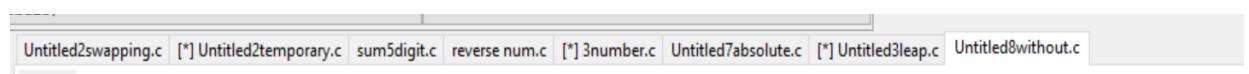
Step 4 :if (year%4==0 && year%100!=0) ? printf("LEAP YEAR") :(year%400 ==0) ?

Step 5 :PRINT "leap year"

Step 6 :else PRINT "common year"

Step 7 :Stop

C code:



```
1 #include<stdio.h>
2 #include<conio.h>
3 int main()
4 {
5     //Declare int year
6
7     int year;
8
9     printf("Enter any year: ");
10    scanf("%d", &year);           //READ year
11
12    (year%4==0 && year%100!=0) ? printf("LEAP YEAR") : //if condition is true than print Leap year
13    (year%400 ==0 ) ? printf("LEAP YEAR") : printf("COMMON YEAR"); //else print common year
14
15
16 }
17
```

Output:

```
C:\Users\Prasad\Documents\Untitled8without.exe
Enter any year: 2021
COMMON YEAR
-----
Process exited after 3.129 seconds with return value 11
Press any key to continue . . .
```

Date:10/9/2020

Exercise 9: Determine whether the character entered is a capital case letter, smaller case letter, a digit or a special symbol using 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
1 #include <stdio.h>
2 #include<conio.h>
3
4 void main()
5 {
6     char ch;           //Declare char 'ch'
7     printf("Enter any character:\n");
8     scanf("%c",&ch);    // Read ch
9     (ch>=65 && ch<=90)?printf("\n Capital Letter"):printf(" ");
10    (ch>=97&&ch<=122)?printf("\nSmall Leter"):printf(" ");
11    (ch>=48&&ch<=57)?printf("\nDigit"):printf(" ");
12    ((ch>=0&&ch<=47)|| (ch>=58&&ch<=64)|| (ch>=91&&ch<=96)|| (ch>=123&&ch<127))?printf("Special Symbol"):printf(" ");
13
14 }
```

Output:

```
Untitled1,9th.exe
C:\Users\Prasad\Documents\Untitled1,9th.exe
Enter any character:
D

Capital Letter
-----
Process exited after 2.906 seconds with return value 1
Press any key to continue . . .
```

```
Untitled1,9th.exe [Executing] Dev C++ 5.11
C:\Users\Prasad\Documents\Untitled1,9th.exe
Enter any character:
9

Digit
-----
Process exited after 3.814 seconds with return value 1
Press any key to continue . . .
```

```
File Search View Project Execute Tools AStyle Window Help
Untitled1,9th.exe
Enter any character:
g

Small Leter
-----
Process exited after 1.38 seconds with return value 1
Press any key to continue . . .
```

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($ch \geq 65 \ \&\ \& ch \leq 90$), then Print capital letter.
 Step 5 : else if($ch \geq 97 \ \&\ \& ch \leq 122$), then Print Small letter
 Step 6 : else if($ch \geq 48 \ \&\ \& ch \leq 57$), then Print Digit
 Step 7 : else if(($ch \geq 0 \ \&\ \& ch \leq 47$) || ($ch \geq 58 \ \&\ \& ch \leq 64$) || ($ch \geq 91 \ \&\ \& ch \leq 96$) || ($ch \geq 123 \ \&\ \& ch \leq 127$)), then Print special symbol
 Step 8 : Stop

C code:

```
Untitled2swapping.c [*] Untitled2temporary.c sum5digit.c reverse num.c [*] 3number.c Untitled7absolute.c [*] Untitled3leap
1 #include<stdio.h>
2 #include<conio.h>
3 void main()
4 {
5     char ch;           //Declare char 'ch'
6     printf("Enter a character:\n");
7     scanf("%c",&ch);    //Read ch
8     if(ch>=65 && ch<=90)      //Conditions are applied and checked,
9     {
10        printf("\n Capital letter"); //if(ch>=65 && ch<=90), then Print capital letter.
11    }
12    else if(ch>=97 && ch<=122)
13    {
14        printf("\n Small letter"); //else if(ch>=97 && ch<=122), then Print Small letter
15    }
16    else if(ch>=48 && ch<=57)
17    {
18        printf("\n Digit"); //else if(ch>=48 && ch<=57), then Print Digit
19    }
20    else if((ch>=0 && ch<=47) || (ch>=58&& ch<=64) || (ch>=91 && ch<=96) || (ch>=123 && ch<=127))
21    {
22        printf("\n Special symbol");
23    }
}
```

Output:

```
C:\Users\Prasad\Documents\char.exe
Enter a character:2020
Digit
-----
Process exited after 2.314 seconds with return value 7
Press any key to continue . . .
```

```
File Search View Project Execute Tools AStyle Window Help
C:\Users\Prasad\Documents\Untitled1,9th.exe
Enter any character:
g
Small Leter
-----
Process exited after 1.38 seconds with return value 1
Press any key to continue . . .
```

```
Untitled,9th
C:\Users\Prasad\Documents\Untitled1,9th.exe
Enter any character:
D
Capital Letter
-----
Process exited after 2.906 seconds with return value 1
Press any key to continue . . .
```

Exercise 11: Check whether a triangle is valid or not

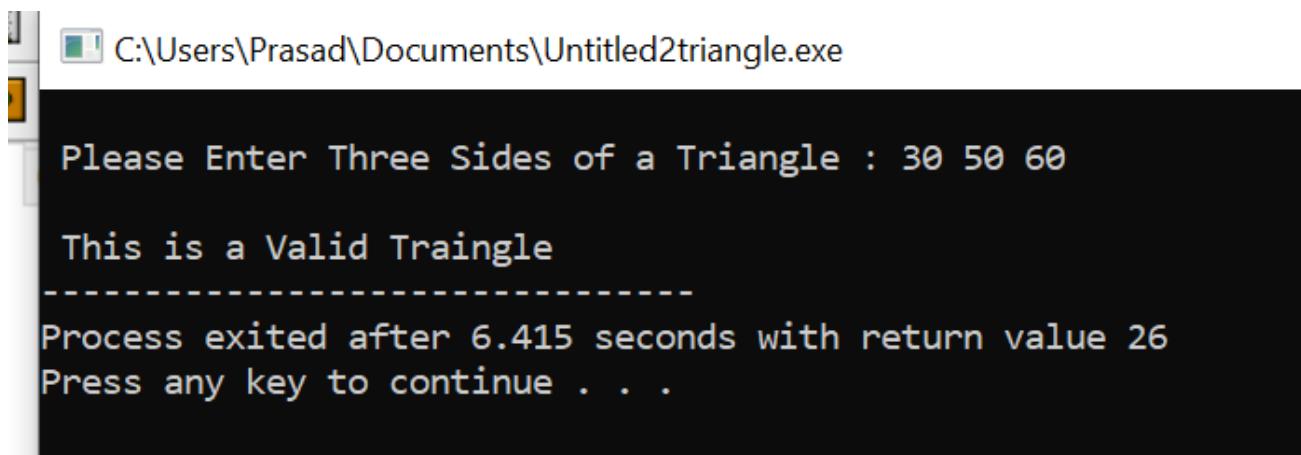
Algorithm:

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 [*] Untitled2triangle.c
1 #include<stdio.h>
2 #include<conio.h>
3 int main()
4 {
5     int g1, g2, g3; //Declare int g1, g2, g3;
6     printf("\n Please Enter Three Sides of a Triangle : ");
7     scanf("%d%d%d", &g1, &g2, &g3); // Read int g1, g2, g3;
8
9     if( (g1 + g2 > g3) && (g2 + g3 > g1) && (g1 + g3 > g2) ) //Conditions are applied and checked,
10    {
11        printf("\n This is a Valid Traingle"); //if (g1 + g2 > g3) && (g2 + g3 > g1) && (g1 + g3 > g2)
12    }
13    else
14    {
15        printf("\n This is an Invalid Triangle"); //Else, then PRINT "This is an Invalid Triangle"
16    }
17
18 }
```

Output:



The screenshot shows a terminal window with the following output:

```
C:\Users\Prasad\Documents\Untitled2triangle.exe
Please Enter Three Sides of a Triangle : 30 50 60
This is a Valid Traingle
-----
Process exited after 6.415 seconds with return value 26
Press any key to continue . . .
```

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:

The screenshot shows a code editor window titled "Untitled1 [*] Untitled4fact.c". The code is as follows:

```
1 #include<stdio.h>
2 #include<conio.h>
3 int main()
4 {
5     int i,g=1,num; //Declare i,g=1,num;
6     printf("Enter a num: "); //Read num
7     scanf("%d",&num);
8     for(i=1;i<=num;i++) //Use for Loop for finding factorial
9     {
10         g=g*i;
11     }
12     printf("Factorial of %d is: %d",num,g);
13 }
14
```

Output:

The screenshot shows a terminal window with the following output:

```
C:\Users\Prasad\Documents\Untitled4fact.exe
Enter a num: 7
Factorial of 7 is: 5040
-----
Process exited after 2.844 seconds with return value 23
Press any key to continue . . .
```

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
1 #include <stdio.h>
2 #include <conio.h>
3 int main()
4 {
5     int i, g1 = 0, g2 = 1, nextnum;           //Take integer variable g1, g2, nextnum
6     printf("Fibonacci Series: ");           //Set g1 = 0, g2 = 1
7
8     for (i = 1; i <= 500; ++i)
9     {
10        printf("%d, ", g1);                 //DISPLAY g1
11        nextnum = g1 + g2;
12        g1 = g2;
13        g2 = nextnum;
14    }
15
16 }
```

Output:

```
C:\Users\Prasad\Documents\Untitled5fabo.exe
Fibonacci Series: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765, 10946, 17711, 28657, 46368, 75025, 121393, 196418, 317811, 514229, 832040, 1346269, 2178309, 3524578, 5702887, 9227465, 14930352, 24157817, 39088169, 63245986, 102334155, 165580141, 267914296, 433494437, 701408733, 1134963170, 1836311903, -1323752223, 512559680, -811192543, -298632863, -1109825406, -1408458269, 1776683621, 368225352, 2144908973, -1781832971, 363076002, -1418756969, -1055680967, 1820529360, 764848393, -1709589543, -944741150, 1640636603, 695895453, -1958435240, -1262539787, 1073992269, -188547518, 885444751, 696897233, 1582341984, -2015728079, -433386095, 1845853122, 1412467027, -1036647147, 375819880, -660827267, -285007387, -945834654, -1230842041, 2118290601, 887448560, -1289228135, -401779575, -1691007710, -2092787285, 511172301, -1581614984, -1070107325, -1869596475, 1445263496, -424332979, 1020930517, 596597538, 1617528055, -2080840975, -90618175, -889489150, -980107325, -1869596475, 1445263496, -424332979, 1020930517, 596597538, 1617528055, -2080841703, -463313648, 1750811945, 1287498297, -1256657054, 30841243, -1225815811, -1194974568, 1874176917, 679202349, -1741588030, -1062385681, 1490993585, 428607904, 1919601489, -1946757903, -27156414, -1973914317, -2001070731, 319982248, -1681088483, -1361106235, 1252772578, -108333657, 1144438921, 1036105264, -2114423111, -1078317847, 1102226338, 23908491, 112613429, 1150043320, -2018789147, -865734491, 626779336, 171044845, 797824181, 968869026, 1766693207, -1559405063, 207288144, -15538248318, 1082513827, -455734491, 626779336, 171044845, 797824181, 968869026, 1766693207, -1559405063, 207288144, -1352116919, -1144828775, 1798021602, 653192827, -1843752867, -1196560040, 1260654389, 70094349, 1330748738, 1400843087, -1563375471, -162532384, -1725907855, -1888440239, 680619202, -1207821037, -527201835, -1735022872, 2032742589, 297719717, -1964504990, -1666785273, 6636770933, -1003108240, -33941287, -1342539447, -1681970654, 1270457195, -411513459, 858943736, 447430277, 1306374013, 1753804290, -1234788993, 519015297, -715773696, -196758399, -912532095, -1109290494, -2021822589, 1163854213, -857968376, 305885837, -552082539, -246196702, -798279241, -1044475943, -1842755184, 1407736169, -435019015, 972717154, 537698139, 1510415293, 2048113432, -736438571, 1311674861, 575236290, 1886911151, -1832819855, 54091296, -1778728559, -1724637263, 791601474, -933035789, -141434315, -1074470104, -1215904419, 2004592773, 788688354, -1501686169, -712997815, 2080283312, 1367285497, -847398487, 519887010, -327511477, 192375533, -135135944, 57239589, -77896355, -20656766, -98553121, -119209887, -217763008, -336972895, -554735903, -891708798, -1446444701, 1956813797, 510369096, -1827784403, -1317415307, 1149767586, -167647721, 982119865, 814472144, 1796592009, -1683903143, 112688866, -1571214277, -1458525411, 1265227608, -193297805, 1071929805, 878632002, 1950561807, -1465773487, 484788320, -980985167, -496196847, -1477182014, -1973378861, 844406421, -1128972440, -284566019, -1413538459, -1698104478, 1183324359, -514780119, 668544248, 153764121, 822308361, 976072482, 1798380843, -1520513971, 277866872, -1242647099, -964780227, 2087539970, 1122759743, -1084667583, 38092160, -1046575423, -1008483263, -20550588686, 1231425347, -823633339, 407792008, -415841331, -8049323, -423890654, -431939977, -855830631, -1287770608, -2143601239, 863595449, -1280005790, -416410341, -1696416131, -2112826472, 485724693, -1627101779, -1141377086, 1526488431, 385111345, 1911599776, -1998256175, -86656399, -2084912574, 2123398323, 38485749, -2133083224, -2094597475, 67286597, -2027310878, -1960024281, 307632137, -1652392144, -1344760007, 1297815145, -46944862, 1250870283, 1203925421, -1840171592, -636246171, 1818549533, 1182303362, -1294114401, -111811039, -1405925440, -151736479, 1371305377, -146431102, 1224874275, 1078443173, -1991649848, -913206675, 1390110773, 476904098, 1867014871, -1951048327, -84033456, -2035081783, -2119115239, 140770274, -1978344965, -1837574691, 479847640, -1358527051, -879479411, 20569060834, 1177481423, -1060525039, 116956384, -943568655, -826612271, -1778189926, 1698174099, -72006827, 1626167272, 1554160445, -1114639579, 439520866, -675118713, -235597847, -910716560, -1146314407, -2057030967, 1091621922, -965409045, 126212877, -839196168, -712983291, -1552179459, 2029804546, 477625087, -1787537663, -1309912576, 1197517057, -112395519, 1085121538, 972726019, 2057847557, -1264393720, 793435837, -470939883, 322513954, -148425929, 174088025, 25662096, 199750121, 225412217, 425162338, 650574555, 1075736893, 1726311448, -1492918955, 233392493, -1259526462, -1026133969, 2009306865, 983172896, -1302487535, -319314639, -1621802174, -1941116813, 732048309, -1209068504, -477020195, -1686088699, 2131858402, 445769703, -1717339191, -1271569488, 1306658617, 34489129, 1340547746, 1375036875, -1579382675, -204345800, -1783728475, -1988074275, 523164546, -1464909729, -941745183, 1888312384, 946567201, -1460087711, -513520510, -1973608221, 1807838565, -165769656, 1642068909, 1476299253, -1176599134, 299700119, -876899015, -577198896, -1454097911, -2031296807, 809572578, -1221724229, -412151651, -1633875880, -2046027531, 615063885, -1430963646, -815899761, 2048103889, 1232204128, -1014659279, 217544849, -797114430, -579569581, -1376684011, -1956253592, 962029693, -994223899, -32194206, -1026418105, -1058612311, -2085030416, 1151324569, -933705847, 217618722, -716087125, -498468403, -1214555528, -1713023931, 1367387837, -345636094, 1021751743, 676115649, 1697867392, -1920984255, -223116863, -2144101118, 1927749315, -216351803, 1711397512, 1495045709, -1088524075, 406521634, -682002441, -275480807, -957483248, -1232964055, 2104519993, 871555938, -1318891365, -447335427, -1766226792, 2081405077,
```

```
C:\Users\Prasad\Documents\Untitled5fabo.exe
Process exited after 0.1932 seconds with return value 2396583362
Press any key to continue
```

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
1 #include<stdio.h>
2 #include<conio.h>
3 #include <math.h>
4 void main()
5 {
6     int i,j;
7     for(i=3;i<=500;i++)
8     {
9         for(j=2;j<i;j++)
10        {
11            if(i%j==0)
12            break;
13            if(j>sqrt(i))
14            {
15                printf("%d ",i);
16                break;
17            }
18        }
19    }
```

Output:

C:\Users\Prasad\Documents\rest.exe

```
3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97 101 103 107 109 113 127 131 137 139 149 151 157 163
167 173 179 181 191 193 197 199 211 223 227 229 233 239 241 251 257 263 269 271 277 281 283 293 307 311 313 317 331 337
347 349 353 359 367 373 379 383 389 397 401 409 419 421 431 433 439 443 449 457 461 463 467 479 487 491 499

Process exited after 0.07554 seconds with return value 0
Press any key to continue . . .
```

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
1 #include<stdio.h>
2 #include<conio.h>
3 int main()
4 {
5     int g1, g2, i, num=1;      //Declare integers g1, g2, i, num=1;
6     printf("Enter any two numbers ");
7     scanf("%d%d",&g1,&g2);      // READ g1 and g2
8     .....                         // Use Loop , for(i=1;i<=g2;i++)
9     for(i=1;i<=g2;i++)
10    {
11        num=num*g1;
12    }
13    printf("%d raised to %d is %d",g1,g2,num);
14
15
16 }
```

Output:

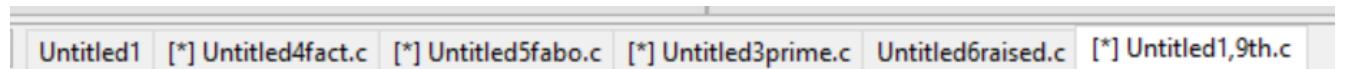
```
C:\Users\Prasad\Documents\Untitled6raised.exe
Enter any two numbers 2 3
2 raised to 3 is 8
-----
Process exited after 5.185 seconds with return value 18
Press any key to continue . . .
```

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
1 #include<conio.h>
2 #include<stdio.h>
3 void main()
4 {
5     int num,r,sum,temp;           //Declare variables num,r,sum,temp
6     printf("Armstrong numbers 1 and 500 are:\n");
7     for(num=1;num<=500;num++)    //Use nested Loop , for(num=1;num<=500;num++)
8     {
9         temp=num;
10        sum=0;
11        while(temp!=0)
12        {
13            r=temp%10;           //while(temp!=0)
14            temp=temp/10;
15            sum=sum+(r*r*r);
16        }
17        if(sum==num)           //use condition if(sum==num)
18        {
19            printf("%d\n",num);
20        }
21    }
22 }
23
```

Output:

```
C:\Users\Prasad\Documents\Untitled1,9th.exe
Armstrong numbers 1 and 500 are:
1
153
370
371
407
-----
Process exited after 0.05953 seconds with return value 125
Press any key to continue . . .
```

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
1 #include <stdio.h>
2 #include<conio.h>
3 int main()
4 {
5     int n, i, k, g = 1;           //Declare int n, i, k, g = 1;
6     printf("Enter the number of rows of Floyd's triangle to print\n");
7     scanf("%d", &n);
8     //Use nested Loop , for(i=1;i<=n;i++)
9     for (i = 1; i <= n; i++)
10    {
11        for (k = 1; k <= i; k++)      //Make inner iteration
12        {
13            printf("%d ",g);          //Print g
14            g++;                     //increment g
15        }
16        printf("\n");
17    }
18 }
```

Output:

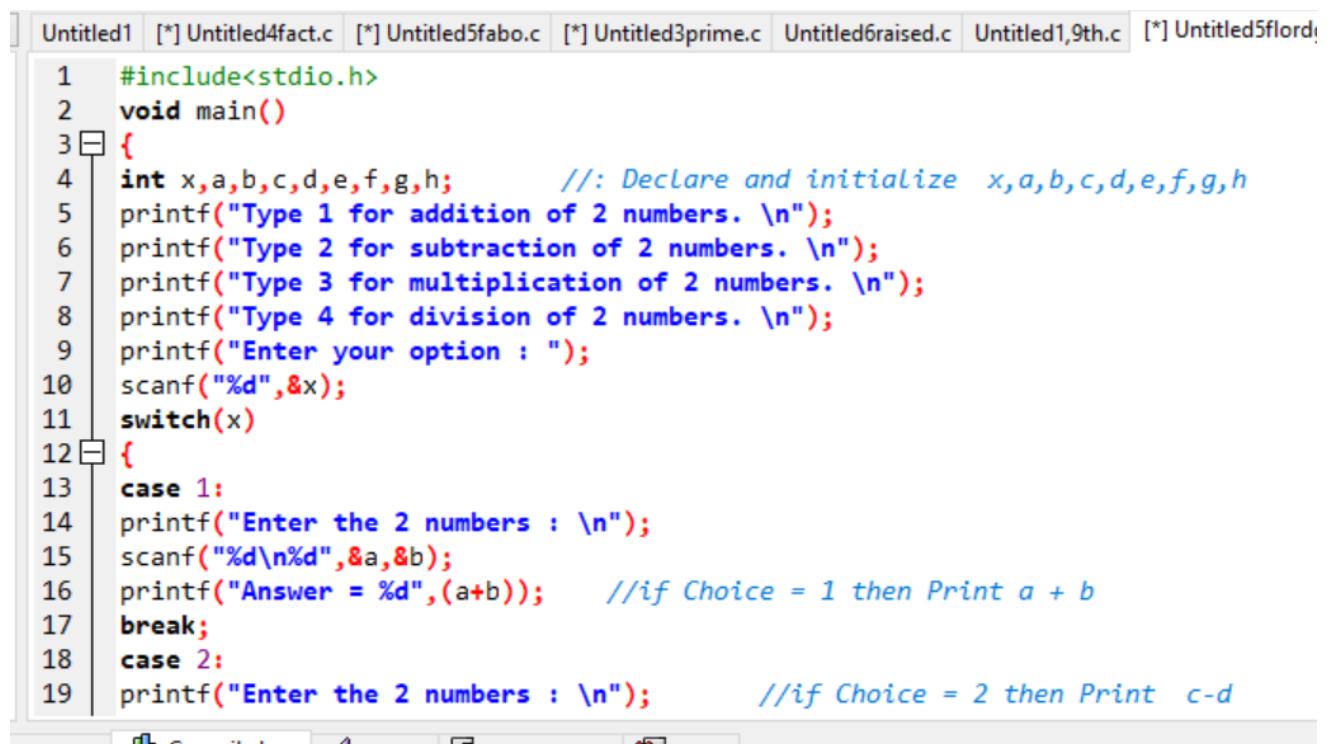
```
C:\Users\Prasad\Documents\Untitled5flordge.exe
Enter the number of rows of Floyd's triangle to print
6
1
2 3
4 5 6
7 8 9 10
11 12 13 14 15
16 17 18 19 20 21
-----
Process exited after 1.941 seconds with return value 6
Press any key to continue . . .
```

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:



```
1 #include<stdio.h>
2 void main()
3 {
4     int x,a,b,c,d,e,f,g,h;      //: Declare and initialize x,a,b,c,d,e,f,g,h
5     printf("Type 1 for addition of 2 numbers. \n");
6     printf("Type 2 for subtraction of 2 numbers. \n");
7     printf("Type 3 for multiplication of 2 numbers. \n");
8     printf("Type 4 for division of 2 numbers. \n");
9     printf("Enter your option : ");
10    scanf("%d",&x);
11    switch(x)
12    {
13        case 1:
14            printf("Enter the 2 numbers : \n");
15            scanf("%d\n%d",&a,&b);
16            printf("Answer = %d", (a+b));    //if Choice = 1 then Print a + b
17            break;
18        case 2:
19            printf("Enter the 2 numbers : \n");    //if Choice = 2 then Print c-d
```

```
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 . . .
```

```
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 : 2
Enter the 2 numbers :
34 9
Answer = 25
-----
Process exited after 6.499 seconds with return value 11
Press any key to continue . . .
```

```
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 : 3
Enter the 2 numbers :
23 76
Answer = 1748
-----
Process exited after 6.246 seconds with return value 13
Press any key to continue . . .
```

```
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 : 4
Enter the 2 numbers :
80 10
Answer = 8
-----
Process exited after 6.856 seconds with return value 10
Press any key to continue . . .
```

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

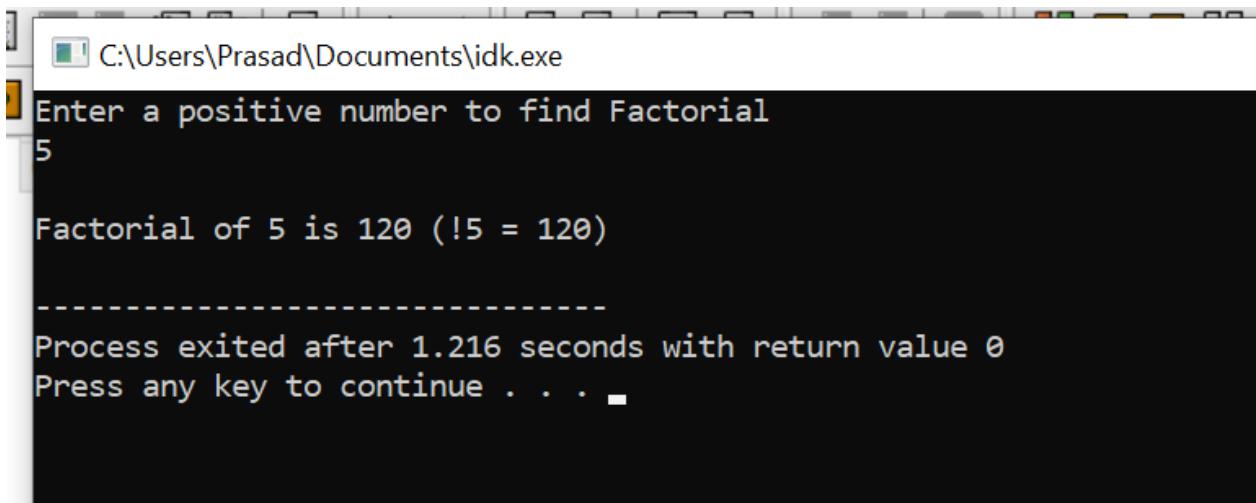
C code:

The screenshot shows a code editor window with a tab bar at the top containing 'Untitled1' and other files like 'Untitled4fact.c', 'Untitled5fabo.c', 'Untitled3prime.c', and 'Untitled6rai'. The main editor area contains the following C code:

```
1 #include<stdio.h>
2
3 void factorial(int);
4
5 int main()
6 {
7     int num;      //Declare num
8     printf("Enter a positive number to find Factorial\n");
9     scanf("%d", &num);      //READ num==0 later
10
11     factorial(num);
12
13     return 0;
14 }
15
16 void factorial(int num)
17 {
18     int count, fact = 1;
19
20     if(num == 0)
21     {
22         printf("Factorial of 0 is 1 (!0 = 1)\n");
23     }
24     else
25     {
```

```
Untitled1 [*] Untitled4fact.c [*] Untitled5fabo.c [*] Untitled3prime.c Untitled6raised.c Untitled1,9th.c [*] Untitled5flordge.c idk.c
10
11     factorial(num);
12
13 }
14
15
16 void factorial(int num)
17 {
18     int count, fact = 1;
19
20     if(num == 0)
21     {
22         printf("Factorial of 0 is 1 (!0 = 1)\n");
23     }
24     else
25     {
26         for(count = 1; count <= num; count++) //use for loop (count = 1; count <= num; count++)
27         {
28             fact = fact * count; //Declare fact
29         }
30     }
31     printf("\nFactorial of %d is %d (!%d = %d)\n", num, fact, num, fact);
32 }
33 }
```

Output:



```
C:\Users\Prasad\Documents\idk.exe
Enter a positive number to find Factorial
5

Factorial of 5 is 120 (!5 = 120)

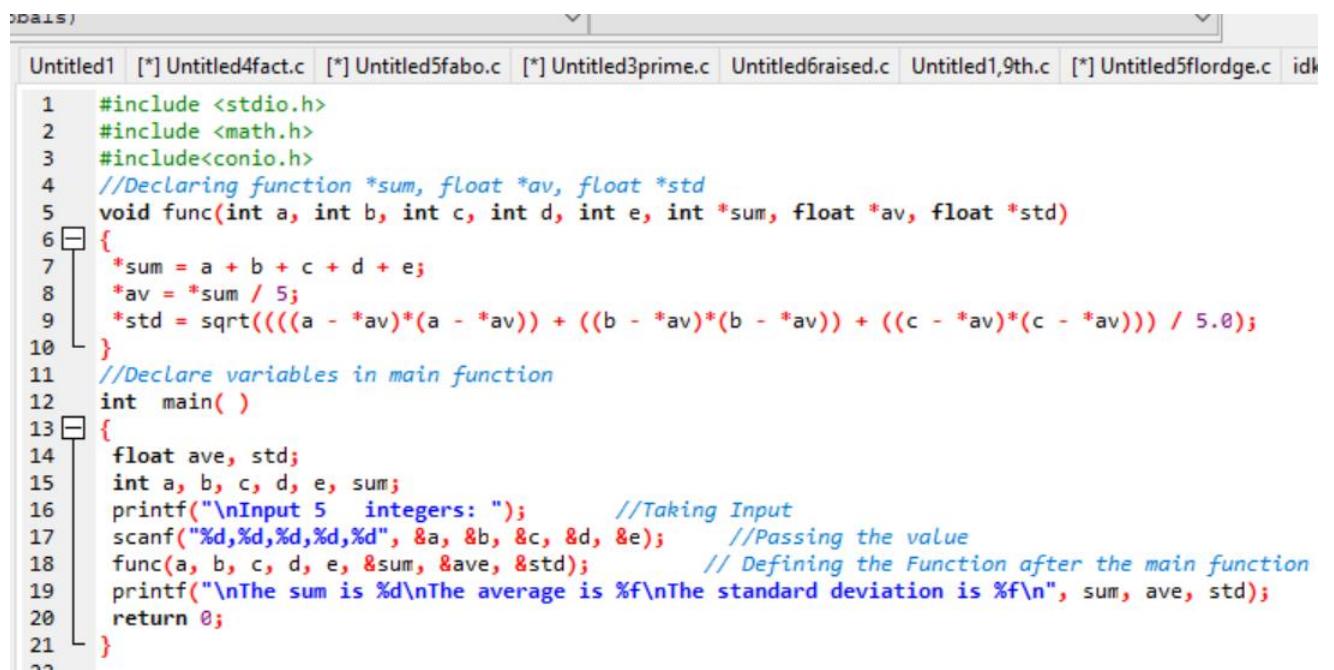
-----
Process exited after 1.216 seconds with return value 0
Press any key to continue . . .
```

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 idk
1 #include <stdio.h>
2 #include <math.h>
3 #include<conio.h>
4 //Declaring function *sum, float *av, float *std
5 void func(int a, int b, int c, int d, int e, int *sum, float *av, float *std)
6 {
7     *sum = a + b + c + d + e;
8     *av = *sum / 5;
9     *std = sqrt(((a - *av)*(a - *av)) + ((b - *av)*(b - *av)) + ((c - *av)*(c - *av))) / 5.0;
10 }
11 //Declare variables in main function
12 int main( )
13 {
14     float ave, std;
15     int a, b, c, d, e, sum;
16     printf("\nInput 5 integers: ");           //Taking Input
17     scanf("%d,%d,%d,%d,%d", &a, &b, &c, &d, &e);      //Passing the value
18     func(a, b, c, d, e, &sum, &ave, &std);        // Defining the Function after the main function
19     printf("\nThe sum is %d\nThe average is %f\nThe standard deviation is %f\n", sum, ave, std);
20     return 0;
21 }
```

Output:

```
C:\Users\Prasad\Documents\Untitled85interger.exe
ss
Input 5 integers: 47893
The sum is 47991
The average is 9598.000000
The standard deviation is 18164.867188
-----
Process exited after 4.483 seconds with return value 0
Press any key to continue . . .
```

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:

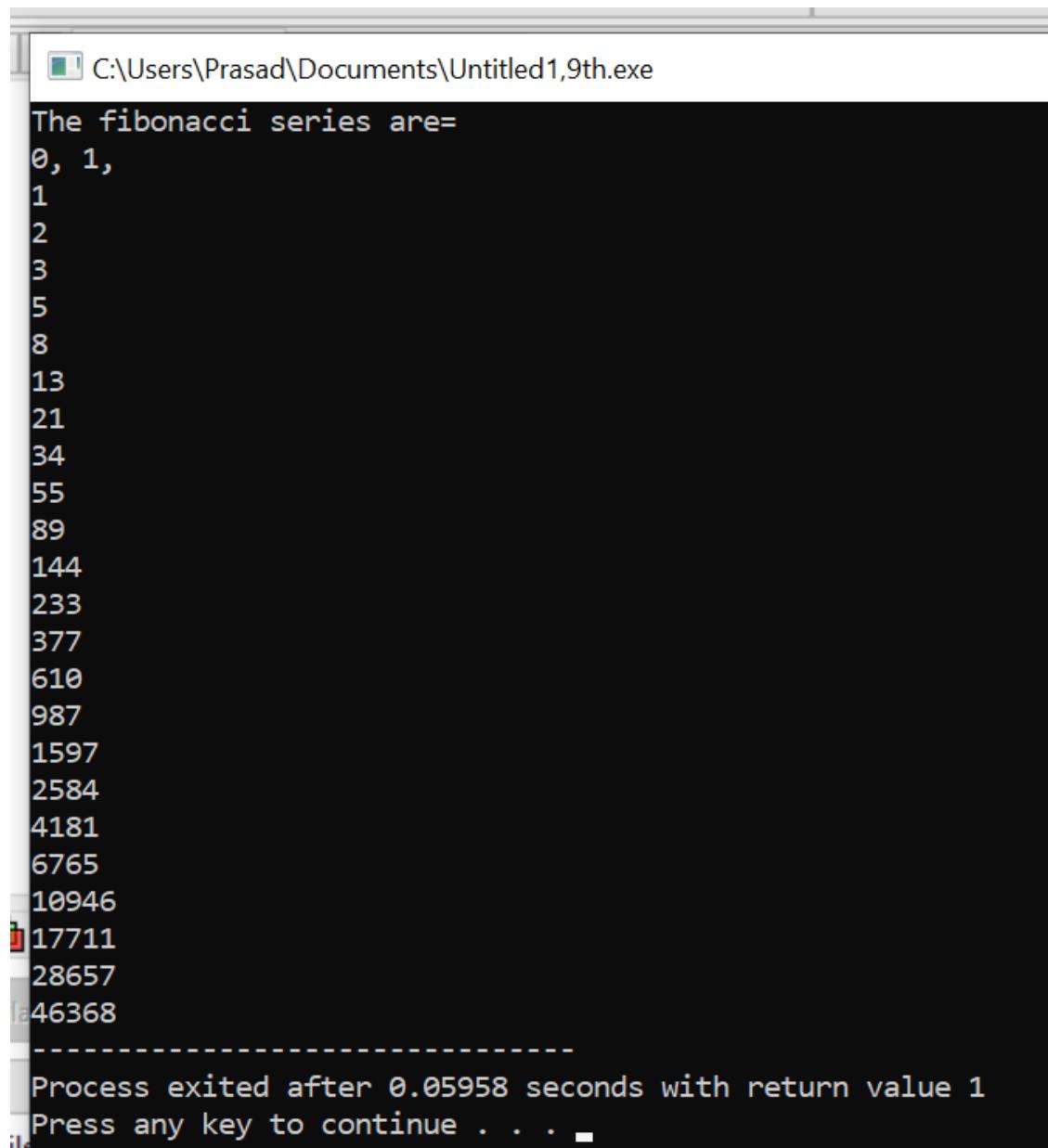
```

Untitled1 [*] Untitled4fact.c [*] Untitled5fabo.c [*] Untitled3prime.c Untitled6raised.c
1 #include<stdio.h>
2 #include<conio.h>
3 long int fibo(long int, long int); //Declaring function fibo
4 void main()
5 //Declare variables in main function
6 {
7 long int i,t,prev=0,curr=1,next;
8 printf("The fibonacci series are=\n"); //Taking Input
9 printf("%ld, %ld, ",prev,curr);
10 fibo(prev,curr);
11 //Passing the values
12 }
13 long int fibo(long int prev, long int curr)
14 //Defining the Function after the main function
15 {
16 static long int count=2;
17 long int nre,next;
18 //By using Loop which runs 25 times
19 if(count<25)
20 {
21 next=prev+curr;
22 printf("\n%ld, ",next); //PRINT the values
23 count=count+1; //Again passing values

```

```
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:



The terminal window shows the output of a C program. The title bar indicates the file path: C:\Users\Prasad\Documents\Untitled1,9th.exe. The program outputs the Fibonacci series starting from 0, followed by a sequence of numbers: 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765, 10946, 17711, 28657, and 46368. A dashed line follows these numbers. Below the dashed line, the terminal displays the message "Process exited after 0.05958 seconds with return value 1" and "Press any key to continue . . .".

```
C:\Users\Prasad\Documents\Untitled1,9th.exe
The fibonacci series are=
0, 1,
1
2
3
5
8
13
21
34
55
89
144
233
377
610
987
1597
2584
4181
6765
10946
17711
28657
46368
-----
Process exited after 0.05958 seconds with return value 1
Press any key to continue . . .
```

Exercise 22:To find binary equivalent of a number

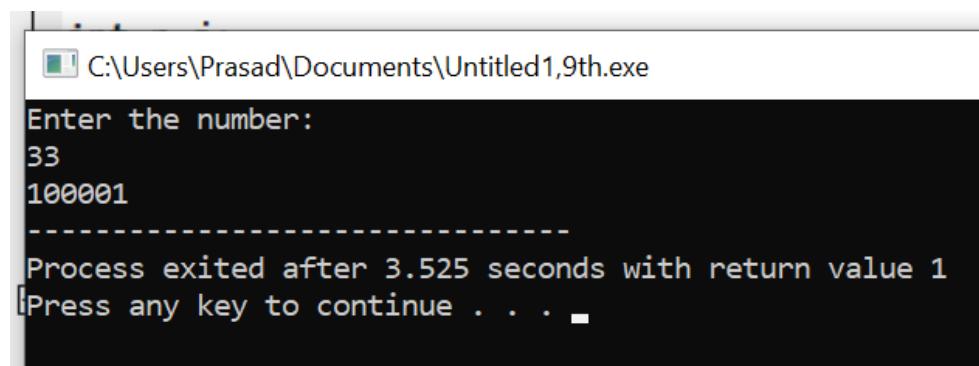
Algorithm:

Step 1 : Start
Step 2 : Declaring the array
Step 3 : Taking Input
Step 4 : By using loops
While($n \neq 0$) : Continue till quotient becomes 0
for($j=i-1; j \geq 0; j--$) : 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
1 #include <stdio.h>
2 void main()
3 {
4     int n,j;
5     int i=0;
6     int a[100]; //Declaring the array
7     printf("Enter the number:\n"); //Taking Input
8     scanf("%d",&n);
9     while(n!=0) //By using loops
10    {           //Continue till quotient becomes 0
11        a[i]=n%2;
12        i++;
13        n=n/2;
14    }
15    for(j=i-1;j>=0;j--) //Printing the array in reverse order to get the binary equivalent
16    {
17        printf("%d",a[j]);
18    }
19 }
```

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 . . .
```

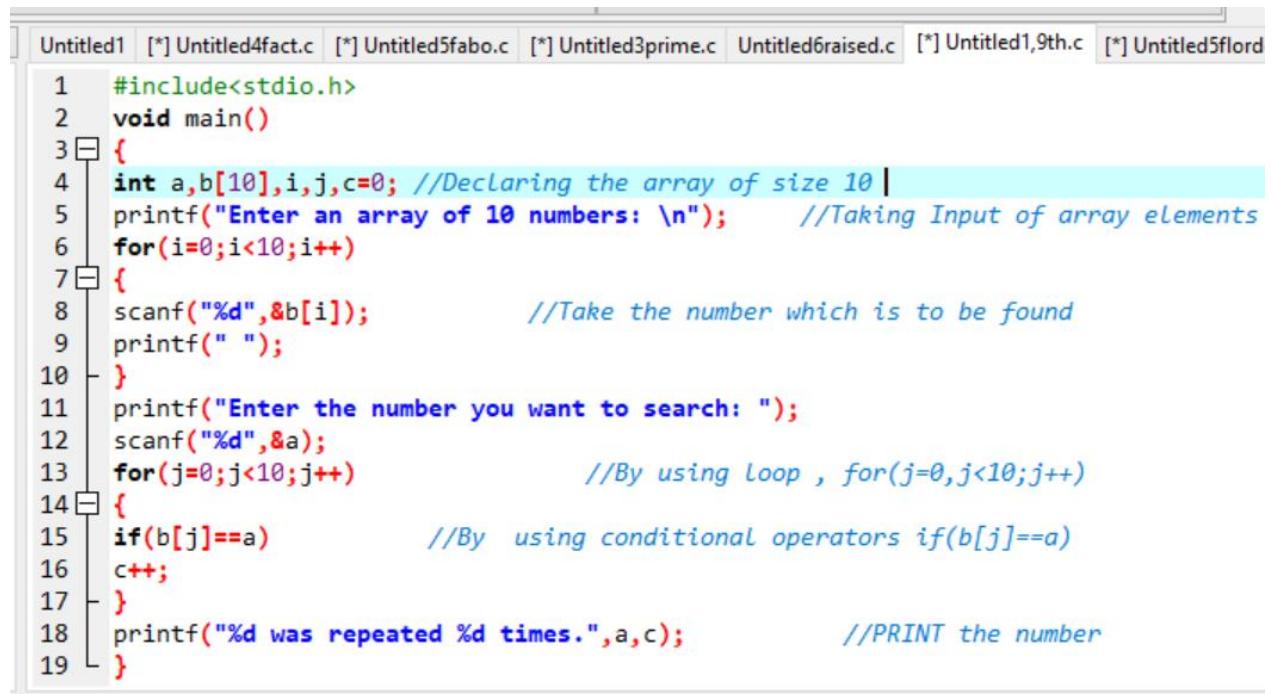
Date:21/9/2020

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
1 #include<stdio.h>
2 void main()
3 {
4     int a,b[10],i,j,c=0; //Declaring the array of size 10 |
5     printf("Enter an array of 10 numbers: \n");      //Taking Input of array elements
6     for(i=0;i<10;i++)
7     {
8         scanf("%d",&b[i]);           //Take the number which is to be found
9         printf(" ");
10    }
11    printf("Enter the number you want to search: ");
12    scanf("%d",&a);
13    for(j=0;j<10;j++)           //By using Loop , for(j=0,j<10;j++)
14    {
15        if(b[j]==a)           //By using conditional operators if(b[j]==a)
16            c++;
17    }
18    printf("%d was repeated %d times.",a,c);      //PRINT the number
19 }
```

Output:

```
C:\Users\Prasad\Documents\Untitled1,9th.exe
Enter an array of 10 numbers:
1
2
3
4
5
7
6
8
9
5
Enter the number you want to search: 4
4 was repeated 1 times.

Process exited after 10.63 seconds with return value 23
Press any key to continue . . .
```

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:

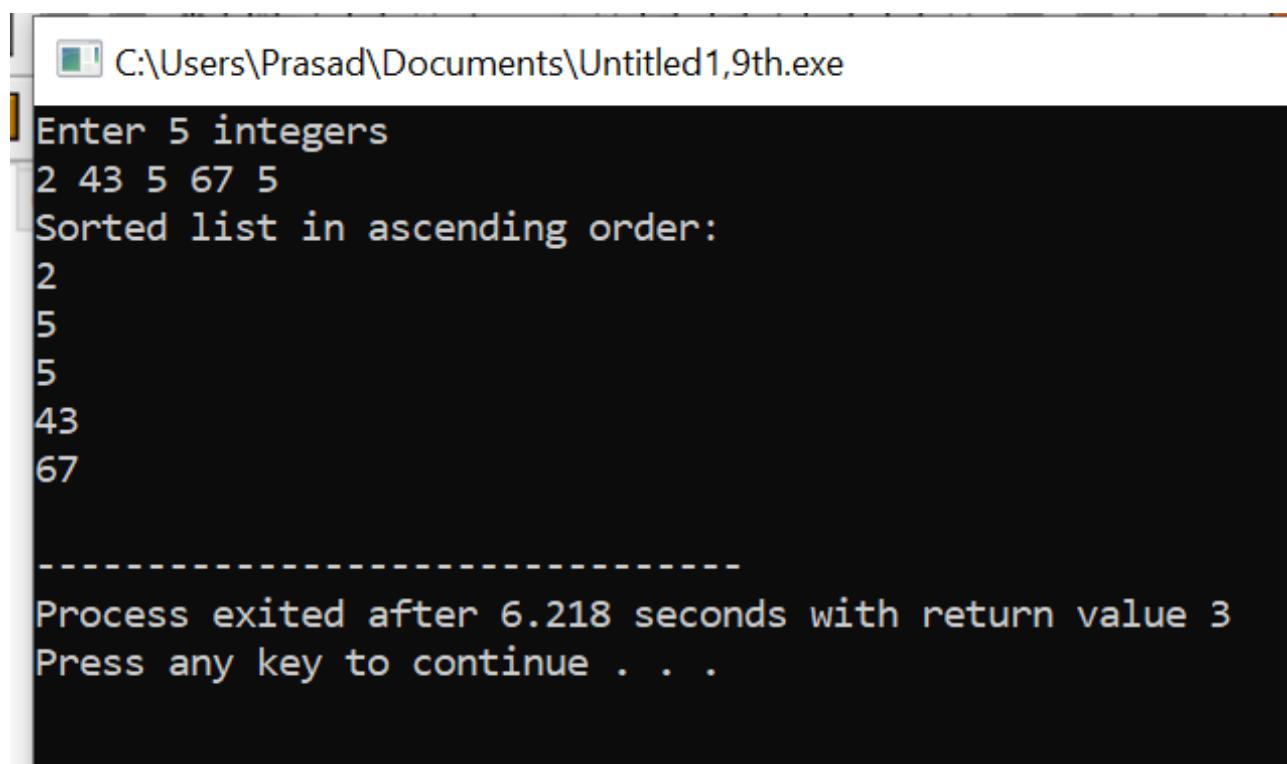
```

1 #include <stdio.h>
2 int main()
3 {
4     int a[5],c,d,p,t;      //Declaring the array of size 5 and other variables
5     printf("Enter 5 integers\n");    //Taking Input of array elements
6     for (c=0;c<5;c++)
7         {
8             scanf("%d",&a[c]);
9         }
10    for (c=0;c<5;c++)
11    {
12        p=c;
13        for (d=c+1;d<5;d++)
14            {
15                if (a[p]>a[d])
16                    p=d;
17            }
18        if(p!=c)
19        {

```

```
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:



```
C:\Users\Prasad\Documents\Untitled1,9th.exe  
Enter 5 integers  
2 43 5 67 5  
Sorted list in ascending order:  
2  
5  
5  
43  
67  
-----  
Process exited after 6.218 seconds with return value 3  
Press any key to continue . . .
```

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:

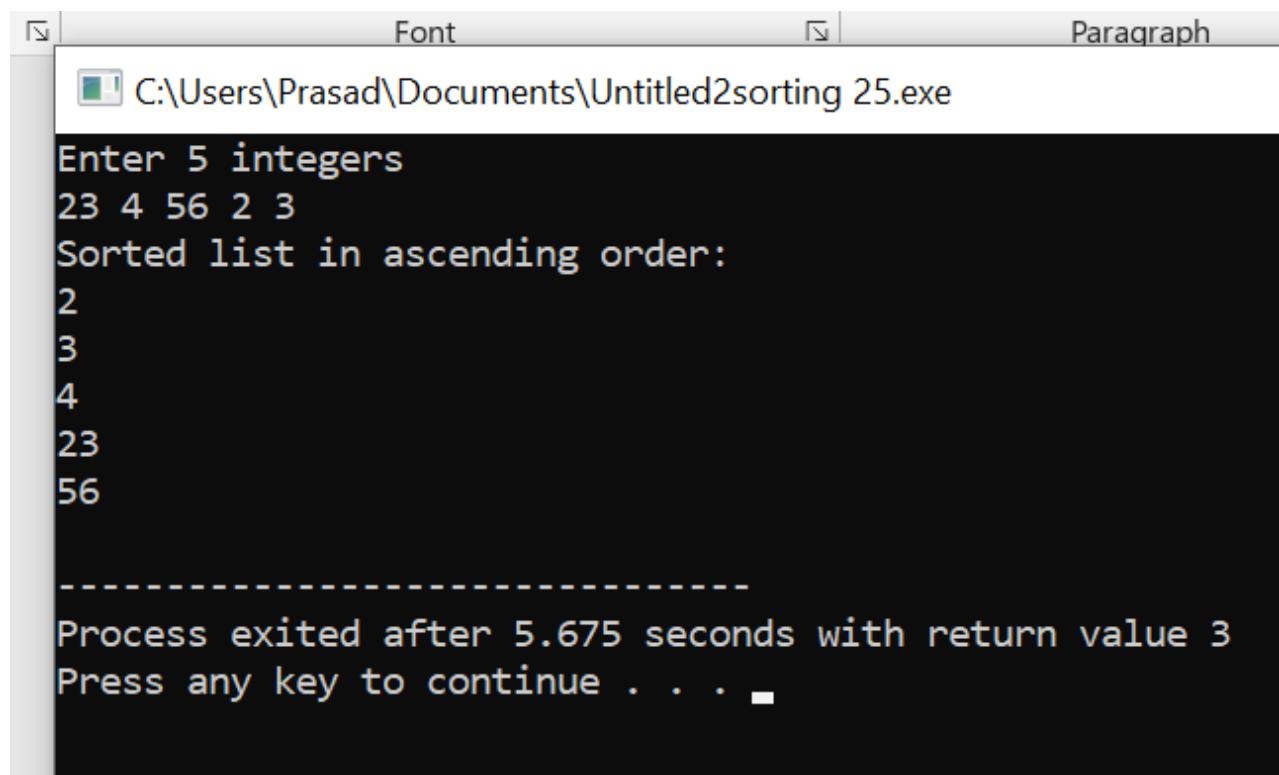
```

Untitled1 [*] Untitled4fact.c [*] Untitled5fabo.c [*] Untitled3prime.c Untitled6raised.c Untitled1,9th.c [ ]
1 #include <stdio.h>
2 int main()
3 {
4     int a[5],c,d,t;
5     //Declaring the array of size 5 and other variables |
6     printf("Enter 5 integers\n");           //Taking Input of array elements
7     for (c=0;c<5;c++)                  //By using nested Loop
8         scanf("%d",&a[c]);
9     for (c=0;c<4;c++)                //For Printing array in ascending order
10    {
11        //using conditional operator
12        for (d=0;d<5-c-1;d++)
13        {
14            //if(a[d]>a[d+1]): Swapping numbers if 1'st value is greater than
15            if(a[d]>a[d+1])
16            {
17                t=a[d];
18                a[d]=a[d+1];
19                a[d+1]=t;

```

```
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 }
27
```

Output:



The screenshot shows a Windows Command Prompt window with the following details:

- WindowTitle: C:\Users\Prasad\Documents\Untitled2sorting 25.exe
- Text:

```
Enter 5 integers
23 4 56 2 3
Sorted list in ascending order:
2
3
4
23
56
```
- Bottom Text:

```
-----
Process exited after 5.675 seconds with return value 3
Press any key to continue . . .
```

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

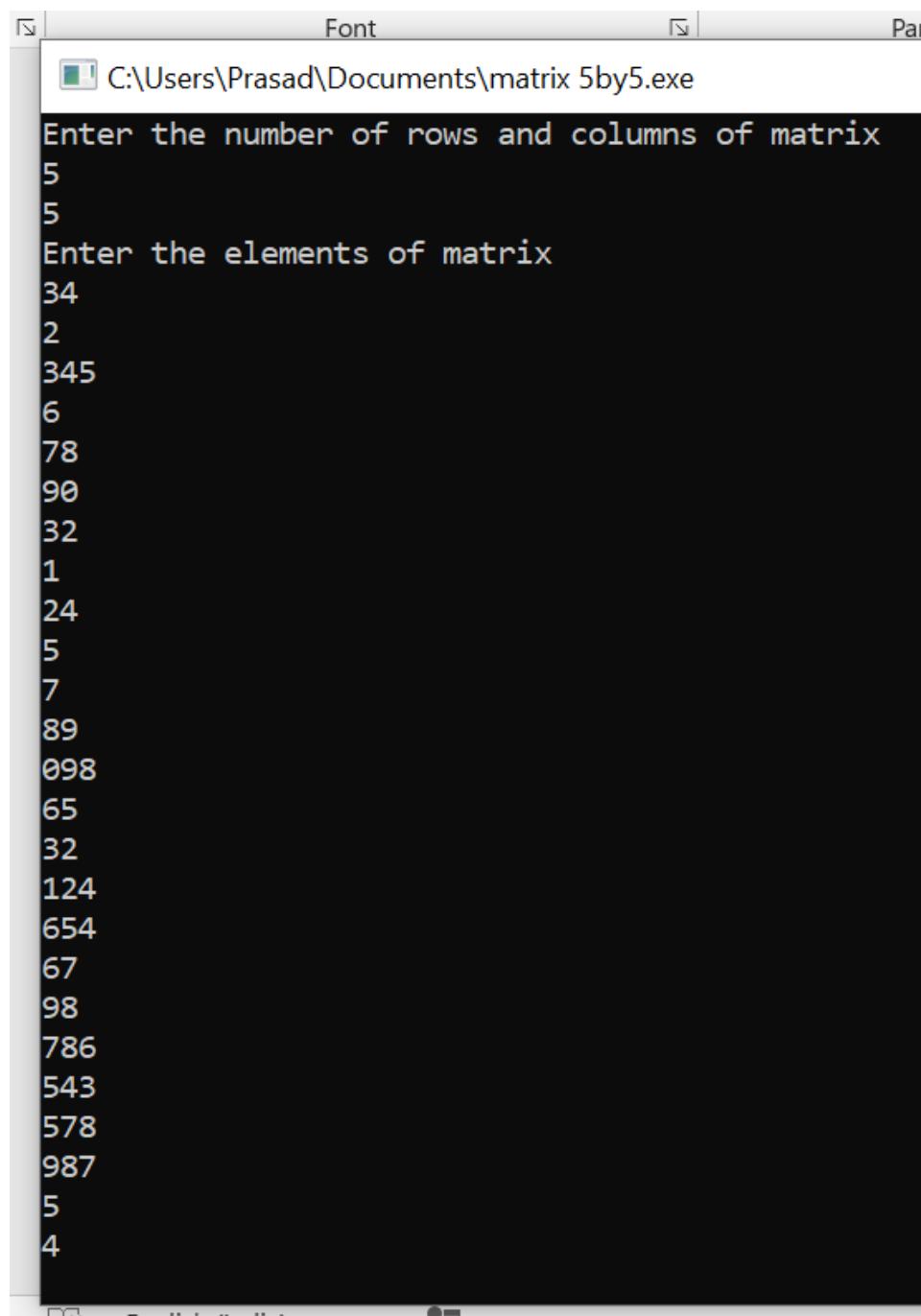
C code:

```
Untitled1 [*] Untitled4fact.c [*] Untitled5fabo.c [*] Untitled3prime.c Untitled6raised.c Untitled1,9th.c [*] Untitled5flordge.c idk.c [*] Untitled8.c
1 #include<stdio.h>
2 #include<conio.h>
3 void main()
4 {
5     int m, n, c, d, matrix[10][10], maximum; //Declaring the array of size and other variable
6     printf("Enter the number of rows and columns of matrix\n"); //Taking Input of array elements
7     scanf("%d%d",&m,&n);
8     printf("Enter the elements of matrix\n");
9     //By using nested Loop , Update Largest number
10    for( c = 0 ; c < m ; c++ )
11    {
12        for( d = 0 ; d < n ; d++ )
13        {
14            scanf("%d",&matrix[c][d]);
15        }
16    }
17    printf("\nMatrix is :\n");
18    for(c=0;c< m;c++)
19    {
```

```
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 }
```

Output:



The screenshot shows a Windows command-line interface window titled 'C:\Users\Prasad\Documents\matrix 5by5.exe'. The window contains the following text:

```
Enter the number of rows and columns of matrix
5
5
Enter the elements of matrix
34
2
345
6
78
90
32
1
24
5
7
89
098
65
32
124
654
67
98
786
543
578
987
5
4
```

```
Font Paragraph  
C:\Users\Prasad\Documents\matrix 5by5.exe  
24  
5  
7  
89  
098  
65  
32  
124  
654  
67  
98  
786  
543  
578  
987  
5  
4  
  
Matrix is :  
34      2      345      6      78  
90      32      1      24      5  
7      89      98      65      32  
124     654      67      98      786  
543     578      987      5      4  
Maximum element in matrix is 987  
-----  
Process exited after 40.25 seconds with return value 33  
Press any key to continue . . .
```

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 loops
- Step 5: PRINT the sum
- Step 6: Stop

C code:

```

Untitled1 3 by 3 matrix.c
1 #include<stdio.h>
2 #include<conio.h>
3
4 int main()
5 {
6     int a[10][10], transpose[10][10], r, c, i, j;
7     printf("Enter rows and columns: ");
8     scanf("%d %d", &r, &c);
9
10    // Assigning elements to the matrix
11    printf("\nEnter matrix elements:\n");
12    for (i = 0; i < r; ++i)
13    {
14        for (j = 0; j < c; ++j) {
15            printf("Enter element a%d%d: ", i + 1, j + 1);
16            scanf("%d", &a[i][j]);
17        }
18
19        // Displaying the matrix a[][]
20        printf("\nEnterd matrix: \n");

```

```
Untitled1 3 by 3 matrix.c
16     }
17
18     // Displaying the matrix a[][]
19     printf("\nEnterd matrix: \n");
20     for (i = 0; i < r; ++i)
21     {
22         for (j = 0; j < c; ++j) {
23             printf("%d ", a[i][j]);
24             if (j == c - 1)
25                 printf("\n");
26         }
27
28         // Finding the transpose of matrix a
29         for (i = 0; i < r; ++i)
30         {
31             for (j = 0; j < c; ++j) {
32                 transpose[j][i] = a[i][j];
33             }
34
35             // Displaying the transpose of matrix a
36             printf("\nTranspose of the matrix:\n");
37         }
38     }
```

```
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     {
30         for (j = 0; j < c; ++j) {
31             transpose[j][i] = a[i][j];
32         }
33
34         // Displaying the transpose of matrix a
35         printf("\nTranspose of the matrix:\n");
36         for (i = 0; i < c; ++i)
37         {
38             for (j = 0; j < r; ++j) {
39                 printf("%d ", transpose[i][j]);
40                 if (j == r - 1)
41                     printf("\n");
42             }
43         }
44     }
45 }
```

Output:

```
C:\Users\Prasad\Documents\3 by 3 matrix.exe
Enter elements for matrix 1:
23
45
6
4
32
1
23
45
6
Enter elemnts for matrix 2:
87
6
56
54
3
4
323
5
76
110 51 62
58 35 5
346 50 82
-----
nProcess exited after 21.85 seconds with return value 10
Press any key to continue . . .
```

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:

```

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 elements 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]);

```

```
Untitled1 3 by 3 matrix.c 3 into 3 multiply.c
14 printf("Enter elements 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 elements 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 . . .
ile
```

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

1 #include<stdio.h>
2 #include<conio.h>
3
4 int main()
5 {
6     int a[10][10], transpose[10][10], r, c, i, j;
7     printf("Enter rows and columns: ");
8     scanf("%d %d", &r, &c);
9
10    // Assigning elements to the matrix
11    printf("\nEnter matrix elements:\n");
12    for (i = 0; i < r; ++i)
13    {
14        for (j = 0; j < c; ++j) {
15            printf("Enter element a%d%d: ", i + 1, j + 1);
16            scanf("%d", &a[i][j]);
17        }
18
19    // Displaying the matrix a[][]
```

```
Untitled1 3 by 3 matrix.c 3 into 3 multiply.c
16         }
17
18     // Displaying the matrix a[][][]
19     printf("\nEnterd 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 3 into 3 multiply.c
27
28
29     // Finding the transpose of matrix a
30     for (i = 0; i < r; ++i)
31         for (j = 0; j < c; ++j) {
32             transpose[j][i] = a[i][j];
33
34     // Displaying the transpose of matrix a
35     printf("\nTranspose of the matrix:\n");
36     for (i = 0; i < c; ++i)
37         for (j = 0; j < r; ++j) {
38             printf("%d ", transpose[i][j]);
39             if (j == r - 1)
40                 printf("\n");
41         }
42 }
```

Output:

```
C:\Users\Prasad\Documents\3 by 3 matrix.exe
$ Enter rows and columns: 3
2

Enter matrix elements:
Enter element a11: 23
Enter element a12: 43
Enter element a21: 21
Enter element a22: 46
Enter element a31: 76
Enter element a32: 55

Entered matrix:
23 43
21 46
76 55

Transpose of the matrix:
23 21 76
43 46 55

-----
Process exited after 13.9 seconds with return value 0
Press any key to continue . . .
```

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
1 #include <stdio.h>
2 #include <string.h>
3 void main()
4 {
5     char a[500]; //Declaring the String of maximum array size 500 and other variables
6     int n,x,i;
7
8     gets(a);
9     printf("From which position the user wants to extract the string?");
10    //Taking Input of string elements : Specific Position
11    scanf("%d",&n);
12    printf("How many characters want to extract?");
13    //Taking Length of string as input
14    scanf("%d",&x);
15    for(i=n;i<=n+x;i++)
16        printf("%c",a[i]);
17        //Printing string from n till return value
18 }
19
```

Output:

```
C:\Users\Prasad\Documents\extractstring.exe
Name Gunjan Deshpande
From which position the user wants to extract the string?5
How many characters want to extract?6
Gunjan
-----
Process exited after 30 seconds with return value 11
Press any key to continue . . . -
```

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
1 #include <stdio.h>
2 #include <string.h>
3 void main()
4 {
5     char names[10][10]; //Declaring the String of maximum array size 10*10 and other variables
6     char t[10];
7     int i,j,x;
8
9     printf("How many names want to enter?"); //Taking names as input from the user
10    scanf("%d",&x);
11    //By using nested loop and conditional operators : name sorting
12    for(i=0;i<x;i++)
13    {
14        scanf("%s",&names[i]);
15    }
16    for(i=0;i<x;i++)
17    {
18        for(j=i+1;j<x;j++)
19        {
```

```
16         for(i=0;i<x;i++)
17         {
18             for(j=i+1;j<x;j++)
19             {
20                 if(strcmp(names[i],names[j])>0)
21                 {
22                     strcpy(t ,names[i]);
23                     strcpy(names[i],names[j]);
24                     strcpy(names[j],t);
25                 }
26             }
27         }
28         printf("\nSorted Names are \n");
29     //Print the names in order
30     for(i=0;i<x;i++)
31         printf("%s\n",names[i]);
32     }
33 }
```

Output:

```
C:\Users\Prasad\Documents\sort names.exe
How many names want to enter?5
Gunjan
Kavita
Johan
Farah
Aqsa

Sorted Names are
Aqsa
Farah
Gunjan
Johan
Kavita

-----
Process exited after 49.78 seconds with return value 5
Press any key to continue . . .
```

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 characters long.**

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

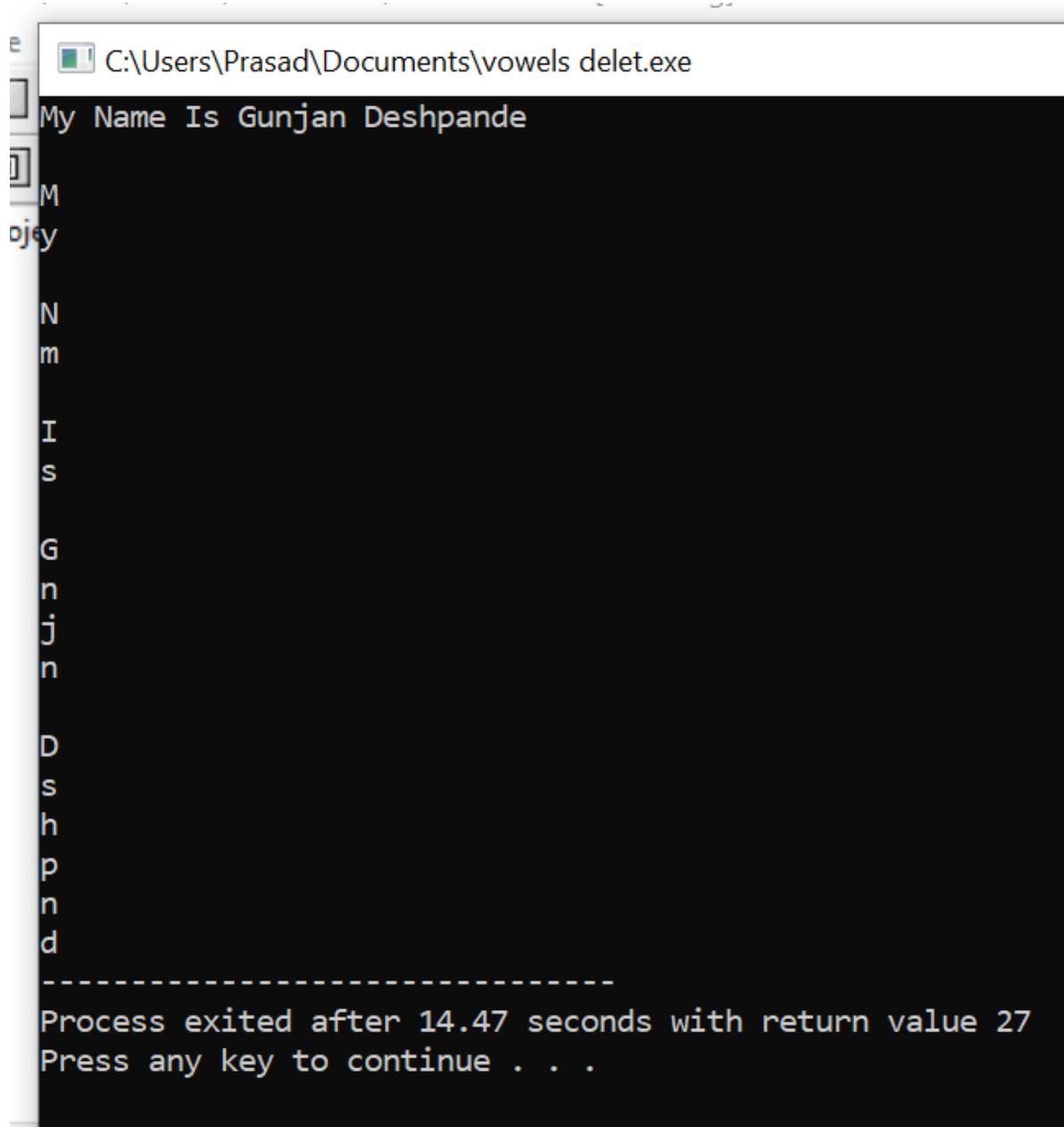
C code:

The screenshot shows a code editor window with a tab bar at the top containing files: Untitled1, 3 by 3 matrix.c, 3 into 3 multiply.c, extractstring.c, sort names.c, and [*] vowels delet.c. The [*] file is currently active. The code itself is as follows:

```
1 #include <stdio.h>
2 #include <string.h>
3 void main()
4 {
5     char a[80];      //Declaring the String of maximum array size 80 and other variables
6     int i;
7     gets(a);        //Take input of String
8     for(i=0;i<strlen(a);i++)
9     {
10         //By using for loop ,conditional operators and logical operators : delete vowels
11         if(a[i]=='a' || a[i]=='e' || a[i]=='i' || a[i]=='o' || a[i]=='u')
12             continue;
13         else
14             printf("\n%c",a[i]); //Print the sentence without the vowels
15     }
16 }
```

The code uses a for loop to iterate through each character of the string. It checks if the character is one of the five vowels ('a', 'e', 'i', 'o', 'u'). If it is, the loop continues to the next character using the `continue` statement. Otherwise, it prints the character to the console using `printf`, effectively removing the vowel from the output.

Output:



C:\Users\Prasad\Documents\vowels delet.exe

```
My Name Is Gunjan Deshpande
M
o
y
N
m
I
s
G
n
j
n
D
s
h
p
n
d
-----
Process exited after 14.47 seconds with return value 27
Press any key to continue . . .
```

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:

```
Untitled1 3 by 3 matrix.c 3 into 3 multiply.c extractstring.c sort names.c [*] vowels del
1 #include<stdio.h>
2 //: Declaring the functions to add book,
3 //display book, display book author , to sort, to link
4 void add_book();
5 void disp_book();
6 void disp_book_auth(int aut_ano);
7 void sortbyano();
8 int sort_function(const void*f,const void*ff);
9 void linkfloat();
10 struct library
11 {
12     //In main function , declare library as structure,
13     //declare arrays for char book-title[20],
14     // char author-name[20] and other variables
15     char book_title[20];
16     char author_name[20];
17     int accno;
18     float price;
19     int flag;
```

```
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;
```

```
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))
```

```
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:");
101    }
```

```
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
```

```
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 void sortbyano()
149 {
150 }
```

```
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
```

Output:

```
C:\Users\Prasad\Documents\library.exe
What do you want to do?
1.Add about a book
2.Display book information
3.List all the book of given author
4.List the title of the specified book
5.List the count of book in library
6.List the book in order of accession number
7.Exit
1
Enter detail of the book
Name of the book
Wings Of Fire
Name of author
Accession number of book
Price of the book
Issued
not issued(0/1):
Book details entered
What do you want to do?
1.Add about a book
2.Display book information
3.List all the book of given author
4.List the title of the specified book
5.List the count of book in library
6.List the book in order of accession number
7.Exit
Enter detail of the book
Name of the book
Name of author
3APJ Abdul Kalam
```

```
C:\Users\Prasad\Documents\library.exe
6.List the book in order of accession number
7.Exit
Enter detail of the book
Name of the book
Name of author
APJ Abdul Kalam
Accession number of book
Price of the book
Issued
not issued(0/1):
Book details entered
What do you want to do?
1.Add about a book
2.Display book information
3.List all the book of given author
4.List the title of the specified book
5.List the count of book in library
6.List the book in order of accession number
7.Exit
Enter detail of the book
Name of the book
Name of author
Accession number of book
231
Price of the book
290
-----
Process exited after 74.2 seconds with return value 3221225477
Press any key to continue . . .
```

return(((struct library*)f)->acno-((struct library*)ff)->acno);

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
1  #include<stdio.h>
2  main()
3  {
4      //Declare files using pointer and other variables
5      FILE *fs;
6      char ch;
7      int i=1;
8      //Opening the file in read only format: fs=fopen("36.c","r");
9      fs=fopen("36.c","r");
10     if(fs==NULL)
11     {
12         printf("can't open source file");
13         exit(1);
14     }
15     //By using conditional operators check whether
16     //If file doesn't exist, then print cannot open source file
17     printf("%d",i++);
18     //By using loop read each character:
19     //use conditional operator , if(ch==EOF), then break, else if(ch=='\n'), new line encountered
```

```
[*] content copy.c line num.c
19 //use conditional operator , if(ch==EOF), then break, else if(ch=='\n'), new line encountered
20 while(1)
21 {
22     ch=fgetc(fs);
23     if(ch==EOF)
24     {
25         break;
26     }
27     printf("%c",ch);
28     //Printing the character
29     if(ch=='\n')
30     {
31         printf("%d",i);
32         i++;
33     }
34 }
35 // Close the file if opened
36 fclose(fs);
37 }
```

Output:

```
C:\Users\Prasad\Documents\line num.exe
1#include<stdio.h>
2#include<conio.h>
3#include<stdlib.h>
4void main()
5{
6    FILE *fptr1,*fptr2;
7    char filename[100],c;
8
9    printf("enter the file name \n");
10   scanf("%s",filename);
11
12   fptr1=fopen(filename,"r");
13   if(fptr1==NULL)
14   {
15       printf("cannot open file %s \n", filename);
16       exit(0);
17   }
18   printf("enter the filename \n");
19   scanf ("%s",filename);
20   fptr2=fopen(filename,"w");
21   if(fptr2==NULL)
22   {
23       printf("cannot open file%s\n",filename);
24       exit(0);
25   }
26   c=fgetc(fptr1);
27   while(c!=EOF)
28   {
29       fputc(c,fptr2);
30       c=fgetc(fptr1);
```

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 format fp1 = 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:

The screenshot shows a C IDE interface with the following details:

- Tab Bar:** content copy .c, line num.c, file end.c, 36.c
- Code Editor:** Displays the following C code:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 int main()
4 {
5     //Declare files using pointer and other variables
6     FILE *fp1, *fp2;
7     char fname1[50], fname2[50], c;
8     //Count for Line number
9     printf("Enter filename to open for reading : ");
10    scanf("%s", fname1);
11    // Open one file for reading
12    fp1 = fopen(fname1, "r");
13    //Opening the file in read only format: fp1 = fopen(fname1, "r");
14    if (fp1 == NULL)
15    {
16        printf("%s file does not exist..", fname1);
17        //By using conditional operators check whether If file doesn't exist
18        // then print file does not exist..
19        exit(0);}
```
- Toolbars:** Resources, Compile Log, Debug, Find Results, Close

```
[*] 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 . . .
```

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 fptra = 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 fptra = 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
1  #include <stdio.h>
2  #include <stdlib.h>
3  int main()
4  {
5      //Declare files and other variables
6      FILE *fptra1, *fptra2;
7      char filename[100], c;
8      //Count for Line number
9      printf("Enter the filename to open for reading \n");
10     scanf("%s", filename);
11
12     // Opening the file for reading fptra1 = fopen(filename, "r");
13     fptra1 = fopen(filename, "r");
14     if (fptra1 == NULL)
15     {
16         printf("Cannot open file %s \n", filename);
17         exit(0);
18     }
19     printf("Enter the filename to open for writing \n"); |
```

```
[*] content copy .c
19 printf("Enter the filename to open for writing \n");
20 scanf("%s", filename);
21
22 // Open another file for writing
23 fptr2 = fopen(filename, "w");
24 if (fptr2 == NULL)
25 {
26     printf("Cannot open file %s \n", filename);
27     exit(0);
28 }
29 //By using Loop read each character:
30 // use operator , while (c != EOF)
31 // Read contents from file
32
33 c = fgetc(fptr1);
34 while (c != EOF)
35 {
36     fputc(c, fptr2);
37     c = fgetc(fptr1);
```

```
[*] content copy .c
29 //By using Loop read each character:
30 // use operator , while (c != EOF)
31 // Read contents from file
32
33 c = fgetc(fptr1);
34 while (c != EOF)
35 {
36     fputc(c, fptr2);
37     c = fgetc(fptr1);
38 }
39 //Copying the contents in 2'nd file using Loop and conditional operator.
40 printf("\nContents copied to %s", filename);
41
42 fclose(fptr1);
43 fclose(fptr2);
44 //Close the files if opened
45 return 0;
46 }
```

Output:

```
C:\Users\Prasad\Documents\content copy .exe
Enter the filename to open for reading
36.c
Enter the filename to open for writing
library.c

Contents copied to library.c
-----
Process exited after 36.18 seconds with return value 0
Press any key to continue . . .
```

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:

```
content copy .c line num.c file end.c 36.c array37.c 37.c [*] 39.c 40.c 38.c 40array.c
1 #include<stdio.h>
2 #include<windows.h>
3 #include<conio.h>
4 #define MAX 5
5 void insert(int*,int pos,int num);
6 void del(int*,int pos);
7 void reverse(int*);
8 void display(int*);
9 void search(int*,int num);
10
11 void main()
12 {
13     int arr[5];
14
15     insert(arr,1,11);
16     insert(arr,2,12);
17     insert(arr,3,13);
18     insert(arr,4,14);
19     insert(arr,5,15);
```

The screenshot shows a Windows-style file explorer window with several files listed in the background. In the foreground, there is a code editor window titled 'array37.c'. The code in the editor is as follows:

```
#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);
```

The code defines five functions: insert, del, reverse, display, and search, along with a main function that initializes an array and calls the insert function five times. The code editor has tabs for 'Resources', 'Compile Log', 'Debug', 'Find Results', and 'Close'.

```
[*] 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);
```

resources 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     {
```

Compile Log Debug Find Results Close

```
locals
[*] 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;
```

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
70 void search(int*arr,int num)
71 {
72     int i;
73     for(i=0;i<MAX;i++)
74     {
75         if(arr[i]==num)
76         {
77             printf("\n\nThe element %d is present at %dth position.",num,i+1);
78             return;
79         }
80     }
81     if(i==MAX)
82     {
83         printf("\n\nThe element %d is not present in the array.",num);
84     }
85 }
86 void display(int*arr)
87 {
88 }
```

```
[*] content copy.c line num.c file end.c 36.c array37.c 37.c [*] 39.c 40.c 38.c 40array.c
79     }
80 }
81 if(i==MAX)
82 {
83     printf("\n\nThe element %d is not present in the array.",num);
84 }
85 }
86 void display(int*arr)
87 {
88     /*traverse the entire array*/
89     int i;
90     printf("\n");
91     for(i=0;i<MAX;i++)
92     {
93         printf("%d\t",arr[i]);
94     }
95 }
96 }
```

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 . . .
```

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
1 #include<stdio.h>
2 #include<conio.h>
3 #include<string.h>
4 #include<windows.h>
5 int xstrlen(char* );
6 void xstrcpy(char*,char* );
7 void xstrcat(char*,char* );
8 int xstrcmp(char*,char* );
9 void show(char* );
10
11 void main()
12 {
13     char s1[]="kicit";
14     char s2[]="Nagpur";
15     char s3[20];
16     int len;
17
18     printf("\nString s1: %s",s1);
19     len=xstrlen(s1);
```

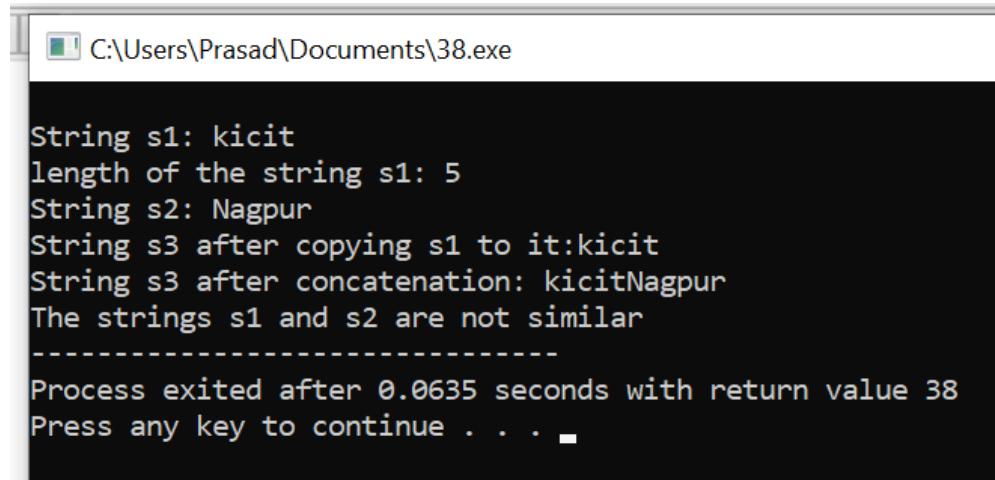
```
[*] content copy .c line num.c file end.c 36.c array37.c 37.c [*] 39.c 40.c 38.c  
16 int len;  
17  
18 printf("\nString s1: %s",s1);  
19 len=xstrlen(s1);  
20 printf("\nlength of the string s1: %d",len);  
21 printf("\nString s2: %s",s2);  
22 xstrcpy(s3,s1);  
23 printf("\nString s3 after copying 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     {
```

```
[*] content copy .c line num.c file end.c 36.c array37.c 37.c [*] 39.c 40.c 38.c  
46 void xstrcpy(char*t,char*s)  
47 {  
48     while(*s)  
49     {  
50         *t=*s;  
51         t++;  
52         s++;  
53     }  
54     *t='\0';  
55 }  
56 void xstrcat(char*t,char*s)  
57 {  
58     while(*t)  
59     {  
60         t++;  
61     }  
62     while(*s)  
63     {  
64         *t++=*s++;  
65     }  
66     *t='\0';  
67 }  
68 int xstrcmp(char*s,char*t)  
69 {  
70     while(*s==*t)  
71     {  
72         if(!(*s))  
73         {  
74             return 0;  
75         }  
76         s++;  
77         t++;  
78     }  
79     return (*s-*t);  
80 }  
81 }
```

```
bals)  
[*] content copy .c line num.c file end.c 36.c array37.c 37.c [*] 39.c 40.c 38.c  
63 {  
64     *t++=*s++;  
65 }  
66     *t='\0';  
67 }  
68 int xstrcmp(char*s,char*t)  
69 {  
70     while(*s==*t)  
71     {  
72         if(!(*s))  
73         {  
74             return 0;  
75         }  
76         s++;  
77         t++;  
78     }  
79     return (*s-*t);  
80 }  
81 }
```

Output:



```
C:\Users\Prasad\Documents\38.exe

String s1: kicit
length of the string s1: 5
String s2: Nagpur
String s3 after copying s1 to it:kicit
String s3 after concatenation: kicitNagpur
The strings s1 and s2 are not similar
-----
Process exited after 0.0635 seconds with return value 38
Press any key to continue . . .
```

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:

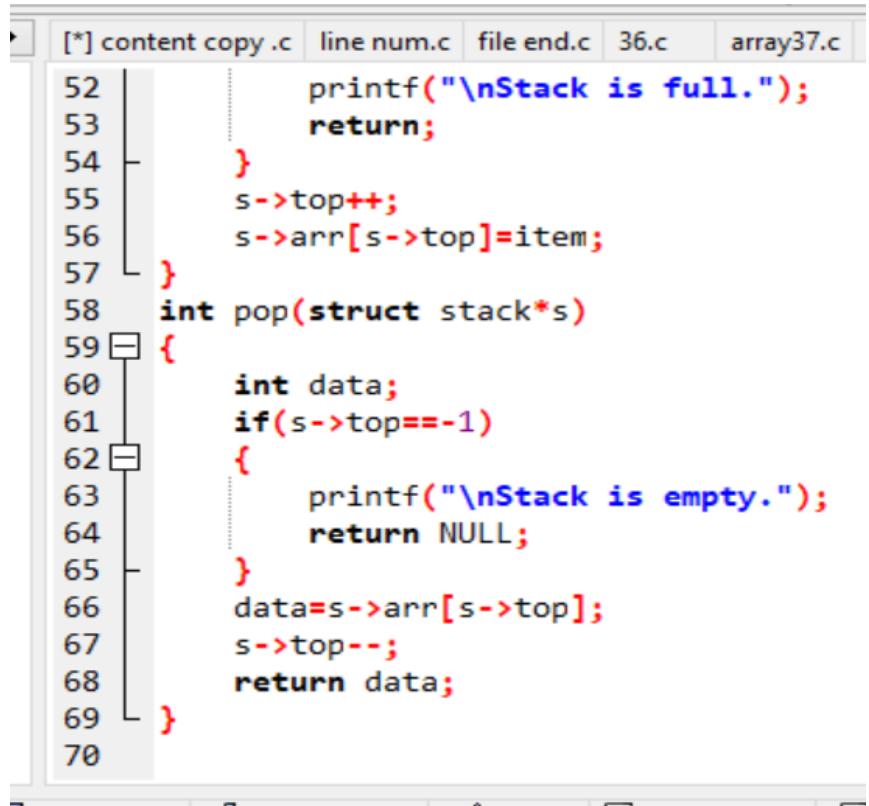
```
[*] content copy .c  line num.c  file end.c  36.c  array37.  
1  #include<stdio.h>  
2  #include<conio.h>  
3  #include<windows.h>  
4  # define MAX 10  
5  struct stack  
6  {  
7      int arr[MAX];  
8      int top;  
9  };  
10 void initstack(struct stack*);  
11 void push(struct stack*,int item);  
12 int pop(struct stack*);  
13  
14 void main()  
15 {  
16     struct stack s;  
17     int i;  
18  
19     initstack(&s);
```

```
[*] content copy .c | line num.c | file end.c | 36.c | array36.c  
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 |



The screenshot shows a code editor window with the following C code:

```
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 . . .
```

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:

```

[*] content copy.c  line num.c  file end.c  36.c  array
1  #include<stdio.h>
2  #define MAX 10
3  #include<stdio.h>
4  #include<conio.h>
5  void addq(int*,int,int*,int*);
6  int delq(int*,int*,int*);
7  void main()
8  {
9      int arr[MAX];
10     int front=-1,rear=-1,i;
11     addq(arr,23,&front,&rear);
12     addq(arr,9,&front,&rear);
13     addq(arr,11,&front,&rear);
14     addq(arr,-10,&front,&rear);
15     addq(arr,25,&front,&rear);
16     addq(arr,16,&front,&rear);
17     addq(arr,17,&front,&rear);
18     addq(arr,22,&front,&rear);
19     addq(arr,19,&front,&rear);

```

Resources Compile Log Debug Find Result

```
[*] 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*prear)
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 Close

The screenshot shows a code editor window with the following C code:

```
46 {  
47     int data;  
48     if(*pfront == -1)  
49     {  
50         printf("\nQueue is Empty.");  
51         return NULL;  
52     }  
53     data = arr[*pfront];  
54     arr[*pfront] = 0;  
55     if(*pfront == *prear)  
56     {  
57         *pfront = *prear = -1;  
58     }  
59     else  
60     {  
61         (*pfront)++;  
62     }  
63     return data;  
64 }
```

The code is part of a function that handles the deletion of an item from a queue. It checks if the queue is empty (line 48). If it is, it prints a message and returns NULL. If the front pointer equals the rear pointer (line 55), it means the queue is empty, so both pointers are set to -1 (line 57). Otherwise, the front pointer is incremented (line 61). Finally, the deleted item is returned (line 63).

Output:

The screenshot shows a terminal window with the following 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 . . .
```

The program outputs "Queue is full." followed by three items being deleted: 23, 9, and 11. A dashed line follows the deletions. The process exits after 0.05662 seconds with a return value of 16. The user is prompted to press any key to continue.

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 evaluated 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 compiling then correct the error or 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.

```

1 #include<stdio.h>
2 int main()
3 {
4     //declare integer i
5     int i,count=0;
6     scanf("%d",&i);
7     //using while loop
8     while(i)
9     {
10         i>=1;
11         count++;
12     }
13     printf("%d\n",count);
14     return 0;
15 }
```

- ❖ 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	

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 $q + \frac{1}{E}$. 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., $\frac{239}{51}$ is equivalent to $239/51 = 4 + \frac{1}{(1+\frac{1}{2}+\frac{1}{(5+\frac{1}{(2+\frac{1}{1}))}))}$. 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 $1 + 2 + 3 = 6$.

```

1 #include<stdio.h>
2 #include<math.h>
3
4 main(){
5     /*perfect number or not*/
6
7     long long g;
8     scanf("%lld",&g);
9     //declare and scan g,
10    //do check the limits of g
11    long long sum_factors = 0;
12    long long temp = (int)sqrt((double)g);
13    long long i;
14    //declare a variable to store
15    //the sum of factors of g and
16    //keep in mind the range g for
17    //choosing the data type for this variable
18
19    for(i=2;i<=temp;i++){
20        if(g%i==0)
21            //check if the sum is equal to g or not
22            sum_factors+=i+(g/i);
23    }
24    if(g>1) //adding 1 to the sum of factors if n>1
25        sum_factors++;
26
27
28
29
30
31
32
33
34
35 }

```

- ❖ After compiling and running the program :

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	

Experiment 3

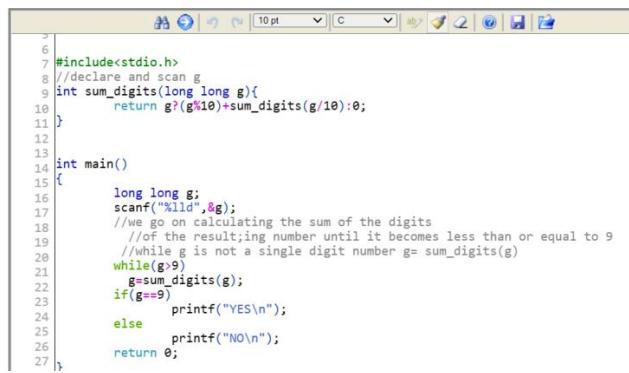
More on Numbers:

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

Programming The 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.



```

2
6 #include<stdio.h>
7 //declare and scan g
8 int sum_digits(long long g){
9     return g?(g%10)+sum_digits(g/10):0;
10 }
11
12
13
14 int main()
15 {
16     long long g;
17     scanf("%lld",&g);
18     //we go on calculating the sum of the digits
19     //of the resulting number until it becomes less than or equal to 9
20     //while g is not a single digit number g= sum_digits(g)
21     while(g>9)
22         g=sum_digits(g);
23     if(g==9)
24         printf("YES\n");
25     else
26         printf("NO\n");
27     return 0;
}

```

- ❖ After compiling and running the program :



Input Data	Expected Output	Code Output	Result	Remarks
1	NO	NO	Passed	
91233256	NO	NO	Passed	
810986905	NO	NO	Passed	
33550336	NO	NO	Passed	
3231	YES	YES	Passed	

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 are 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 x^n 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.

```
1
2
3 #include <stdio.h>
4 #include <math.h>
5 main(){
6     int g,i;
7     double result = 0;
8     scanf("%d",&g);
9
10    for(i=1;i<=g;i++)
11        result+=log10(i);
12
13    printf("%d\n",(int)result+1);
14    return 0;
15 }
```

- ❖ After compiling and running the program :

Language	ProblemNo			
C (gcc 4.3.2)	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

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 whether it is read from start to end or 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.

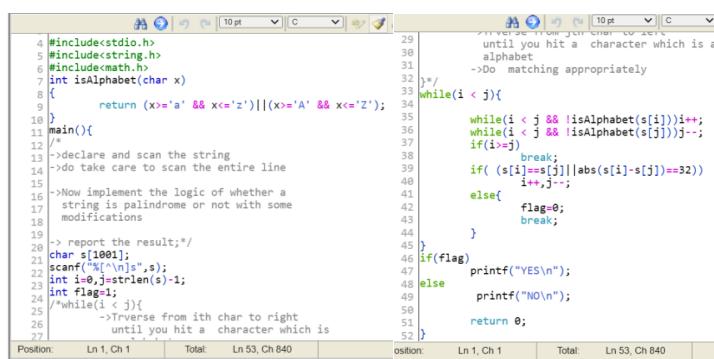
Alternatively, one can create two indices and put them at the end and beginning of the string. If a character at any of the indices is not an alphabet the corresponding indice is moved until an alphabet is encountered. If the alphabet is in Upper case, then it is converted into lower

case. Now, the values at the indices are compared and if they are not same then the string is not a palindrome. Otherwise, the check is repeated for new values of indices obtained by moving them by sufficient 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.



```
4 #include<stdio.h>
5 #include<string.h>
6 #include<math.h>
7 int isAlphabet(char x)
8 {
9     return (x>='a' && x<='z')||(x>='A' && x<='Z');
10 }
11 main()
12 {
13     /*declare and scan the string
14     -do take care to scan the entire line
15     */
16     /*Now implement the logic of whether a
17      string is palindrome or not with some
18      modifications
19      */
20     /* report the result*/
21     char s[100];
22     scanf("%[^\\n]",s);
23     int i=0,j=strlen(s)-1;
24     int flag=1;
25     /*while(i < j){
26         ->Traverse from ith char to right
27         until you hit a character which is
28     }*/
29     /* until you hit a character which is a
30     alphabet
31     ->Do matching appropriately
32     */
33     while(i < j){
34         while(i < j && !isAlphabet(s[i]))i++;
35         while(i < j && !isAlphabet(s[j]))j--;
36         if(i>j)
37             break;
38         if( (s[i]==s[j])||abs(s[i]-s[j])==32)
39             i++,j--;
40         else
41             flag=0;
42         }
43         break;
44     }
45     if(flag)
46         printf("YES\\n");
47     else
48         printf("NO\\n");
49     return 0;
50 }
51
52 }
```

- ❖ After compiling and running the program



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. So in 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).

```

3 #include<stdio.h>
4 int g;
5 int no_ways(int W,int index)
6 {
7     /*->termination conditions
8      ->for each possibility
9       of choosing i items of weight
10      2^index find the number of solutions
11
12     ->return the total number of solutions*/
13
14     if(!W)
15         return 1;
16     if(index==g)
17         return 0;
18
19     int ans = 0,i;
20     for(i=0;i*(1<<index)<=W;i++)
21         ans+=no_ways(W-i*(1<<index),index+1);
22
23     return ans;
24 }

```

position: Ln 22, Ch 40 Total: Ln 33, Ch 434

- ❖ After compiling and running the program

Language	Promocode	Compile	Run	
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	

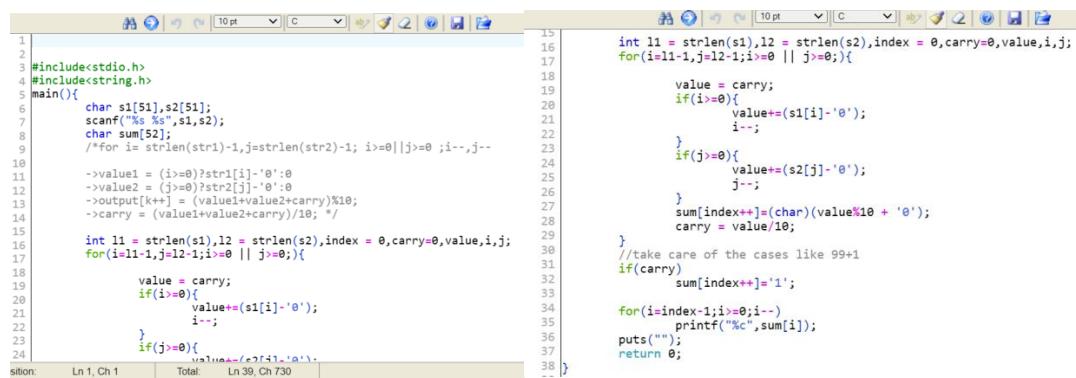
Experiment 7

Advanced Arithmetic:

The main aim of this experiment is to learn to solve problems related to Advanced Arithmetic 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 too big to fit into standard data types, then we can use strings to store the number. So in this case, addition of two numbers 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

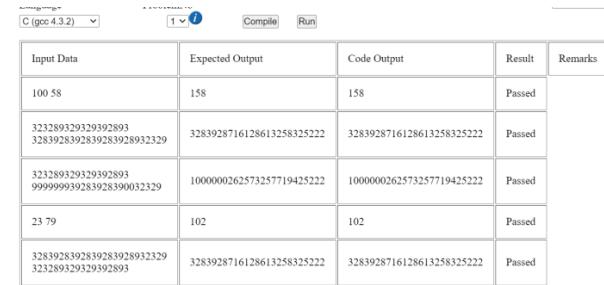


```

1 #include<stdio.h>
2 #include<string.h>
3 main(){
4     char s1[51],s2[51];
5     scanf("%s %s",s1,s2);
6     char sum[52];
7     /*for i= strlen(str1)-1,j=strlen(str2)-1; i>=0 | j>=0 ;i--,j--
8      ->value1 = (i>=0)?str1[i]:'0';
9      ->value2 = (j>=0)?str2[j]:'0';
10     ->output[i+j] = (value1+value2+carry)%10;
11     ->carry = (value1+value2+carry)/10; */
12
13     int l1 = strlen(s1),l2 = strlen(s2),index = 0,carry=0,value,i,j;
14     for(i=l1-1,j=l2-1;i>=0 || j>=0){
15         value = carry;
16         if(i>=0){
17             value+=(s1[i]-'0');
18             i--;
19         }
20         if(j>=0){
21             value+=(s2[j]-'0');
22             j--;
23         }
24         if(j>=0){value+=('0'+l1-i-j-1));
25         }
26         if(i>=0){value+=('0'+l2-j-1));
27         }
28         value+=carry;
29         if(i>=0){
30             value+=(s1[i]-'0');
31             i--;
32         }
33         if(j>=0){
34             value+=(s2[j]-'0');
35             j--;
36         }
37         sum[index++]=value%10 + '0';
38         carry = value/10;
39     }
40     //take care of the cases like 99+1
41     if(carry)
42         sum[index++]='1';
43
44     for(i=index-1;i>=0;i--)
45         printf("%c",sum[i]);
46     puts("");
47     return 0;
48 }

```

- ❖ After compiling and running the program



Input Data	Expected Output	Code Output	Result	Remarks
100 58	158	158	Passed	
32328932932932893 3283928392839283928932329	3283928716128613258325222	3283928716128613258325222	Passed	
32328932932932893 99999993283928390032329	1000000262573257719425222	1000000262573257719425222	Passed	
23 79	102	102	Passed	
328392839283928392893 32328932932932893	3283928716128613258325222	3283928716128613258325222	Passed	

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 problems are 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 evaluated 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 triangles are considered to be valid triangles.
- ❖ 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 $\text{arr}[i]+\text{arr}[j]<\text{arr}[k]$
 7. print the final answer

```

1 //include<stdio.h>
2 //include<stdlib.h>
3 int compare(const void *a,const void *b){
4     return ((int *)a)>((int *)b);
5 }
6 int search(int a,int start,int end,int num){
7     if(start>end)
8         return 0;
9     int mid = (start+end)>>1 ;
10    if(a[mid]==num)
11        return (end-mid+1)+search(a,start,mid-1,num);
12    else
13        return search(a,mid+1,end,num);
14 }
15 main()
16 /*>declare and scan N
17 /*>declare and scan the array
18 /*>sort the array
19 /*>for(i=0;i< N;i++)
20    /*>for(j=i+1;j< N;j++)
21       /*>for(k=j+1;k< N;k++)
22          ans+=search(arr,0,N-1,arr[i]+arr[j]);
23
24 */->print the final answer
25 */
26 {
27     int N;
28     scanf("%d",&N);
29     int arr[N];
30     int i,j;
31     for(i=0;i< N;i++)
32         scanf("%d",&arr[i]);
33     qsort(arr,N,sizeof(int),compare);
34     int ans=0;
35     for(i=0;i< N;i++)
36         for(j=i+1;j< N;j++)
37             ans+=search(arr,0,N-1,arr[i]+arr[j]);
38     printf("%d\n",ans);
39     return 0;
40 }

```

Input Data	Expected Output	Code Output	Result	Remarks
10 50 64 14 87 18 57 13 12 29 36	69	69	Passed	
10 50 41 46 5 92 82 18 13 78 65	60	60	Passed	
10 29 85 82 89 19 52 74 97 2 89	51	51	Passed	
10 63 46 96 71 40 84 49 52 52 65	0	0	Passed	
10 5 76 80 72 31 38 27 88 55 84	34	34	Passed	

- ❖ After compiling and running the program

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 distinct 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.

```

1  #include<stdio.h>
2
3  int func(int N,int *weight,int S){
4      if(!S)// If target weight has been achieved, we have found a solution.
5          return 1;
6
7      if(N) // We have exhausted all the weights
8          return 0;
9      /* Case1: We consider the weight for the solution
10     Case2: We discard the weight for the solution
11 */
12     return func(N-1,weight,S-weight[N-1])+func(N-1,weight,S);
13 }
14 main(){
15     int N;
16     scanf("%d",&N);
17     int weight[N];
18     // weight[i] stores the respective weights
19     for(int i=0;i<N;i++)
20         scanf("%d",&weight[i]);
21     int Sum;
22     scanf("%d",&Sum);
23     int no_ways = func(N,weight,Sum);
24     if(no_ways<0)
25         no_ways = -1;
26     printf("%d\n",no_ways);
27 }
28

```

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

❖ After compiling and running the program

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	

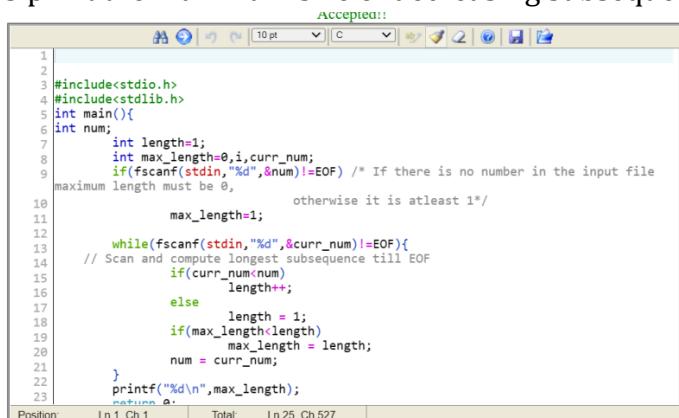
Experiment 10

Sequences:

This experiment is nothing but just an introduction 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 compiling then correct the error or 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 the maximum size of the continuous subsequence and the size of current decreasing subsequence
 4. Iterate over the array and update the variables
 5. print the maximum size of decreasing subsequence.



```

1 //include<stdio.h>
2 //include<stdlib.h>
3 int main(){
4     int num;
5     int length=1;
6     int max_length=0,i,curr_num;
7     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*/
8         max_length=1;
9     while(fscanf(stdin,"%d",&curr_num)!=EOF){
10        // Scan and compute longest subsequence till EOF
11        if(curr_num>num)
12            length++;
13        else
14            length = 1;
15        if(max_length<length)
16            max_length = length;
17        num = curr_num;
18    }
19    printf("%d\n",max_length);
20    return 0;
21 }
22
23

```

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	

- ❖ After compiling and running the program

