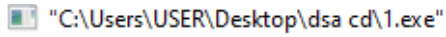
Experiment No: 1 **Date:** 04 - 02 - 2023

Experiment Name: Write a program to sort some data using Bubble sort.

Code:

```
// Bubble Sort
#include <stdio.h>
int main()
{
   int array[1000],size, swap;
   printf("Bubble  Sort\n");
   printf("Enter number of elements: ");
   scanf("%d", &size);
   printf("\nEnter %d integers\n", size);
   for (int step = 0; step < size; step++)
      scanf("%d", &array[step]);
   for ( int step = 0 ; step < size - 1; step++)
   {
      for (int i = 0 ; i < size - step - 1; i++)
      {
         if (array[i] > array[i+1])
         {
            swap       = array[i];
            array[i] = array[i+1];
            array[i+1] = swap;
         }
      }
   }
   printf("\nSorted list in ascending order:\n");
   for (int step = 0; step < size; step++)
      printf("%d\n", array[step]);
   return 0;
}
```

Output:




```
Bubble Sort
Enter number of elements: 10

Enter 10 integers
2
4
6
8
-3
-5
-7
0
1
-12

Sorted list in ascending order:
-12
-7
-5
-3
0
1
2
4
6
8

Process returned 0 (0x0)   execution time : 48.714 s
Press any key to continue.
```

Experiment No: 2 **Date:** 04 - 02 - 2023

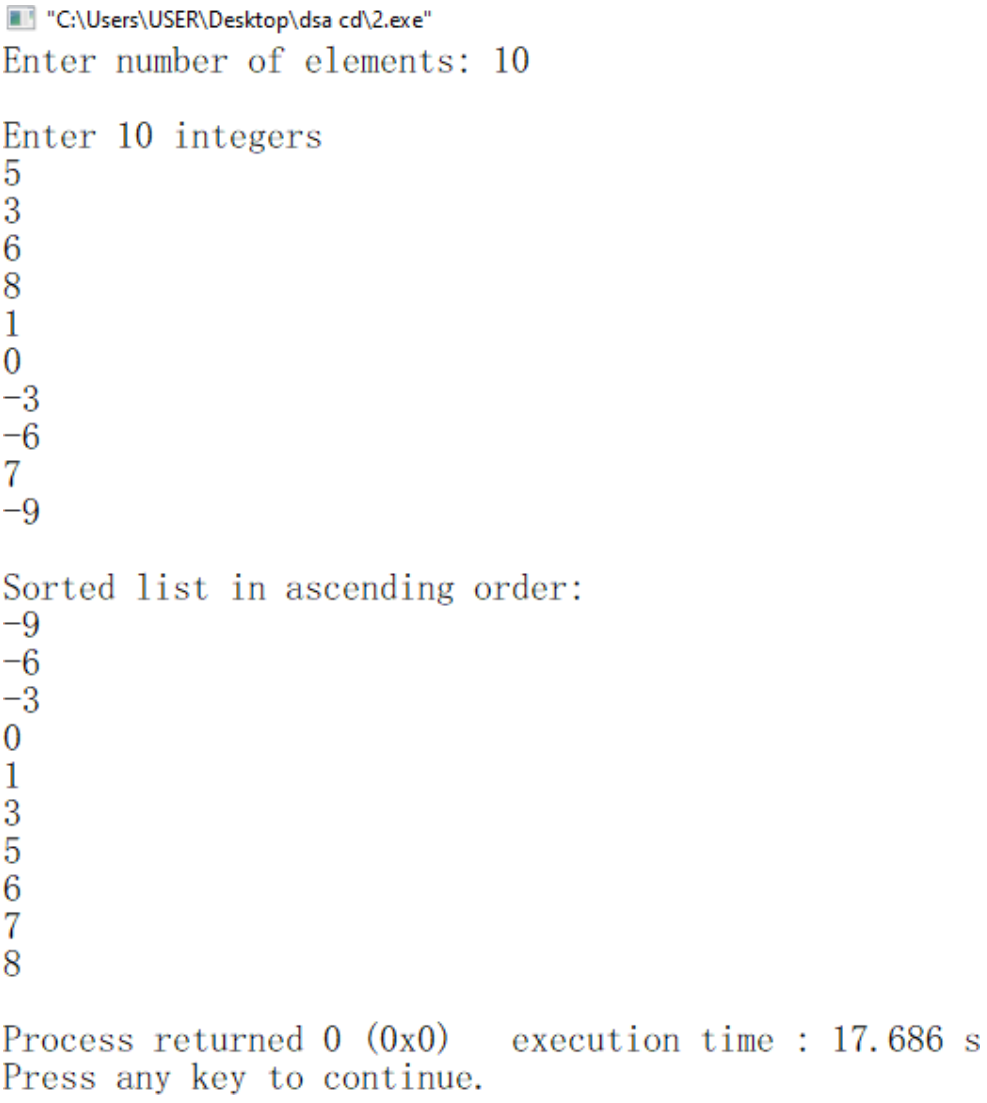
Experiment Name: Write a program to sort some data using Insertion sort.

Code:

```
// Insertion Sort
#include <stdio.h>
int main()
{
   int size, i, temp;
   int array[1000];
   printf("Enter number of elements: ");
   scanf("%d", &size);
   printf("\nEnter %d integers\n", size);
   for (int step = 0; step < size; step++)
   {
      scanf("%d", &array[step]);
   }
   for (int step = 1 ; step <= size - 1; step++)
   {
      i = step;
      while ( i > 0 && array[i-1] > array[i])
      {
         temp        = array[i];
         array[i]    = array[i-1];
         array[i-1]  = temp;
         i--;
      }
   }
   printf("\nSorted list in ascending order:\n");
   for (int step = 0; step <= size - 1; step++)
   {
      printf("%d\n", array[step]);
```

```
    }
    return 0;
}
```

Output:



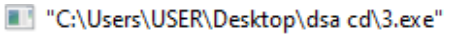
Experiment No: 3 **Date:** 11 - 02 - 2023

Experiment Name: Write a program to search specific data from a list using linear search.

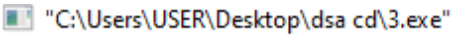
Code:

```
// Linear Search
#include <stdio.h>
int main()
{
    int array[1000], search, i, num, count = 0;
    printf("Enter number of elements in array: ");
    scanf("%d", &num);
    printf("\nEnter %d numbers\n", num);
    for (i = 0; i < num; i++)
        scanf("%d", &array[i]);
    printf("\nEnter a number to search: ");
    scanf("%d", &search);
    for (i = 0; i < num; i++)
    {
        if (array[i] == search)
        {
            printf("%d is present at location %d.\n", search, i+1);
            count++;
        }
    }
    if (count == 0)
        printf("%d isn't present in the array.\n", search);
    else
        printf("\n%d is present %d times in the array.\n", search, count);
    return 0;
}
```

Output:




```
Enter number of elements in array: 6

Enter 6 numbers
3
6
4
8
-1
3

Enter a number to search: 3
3 is present at location 1.
3 is present at location 6.

3 is present 2 times in the array.

Process returned 0 (0x0)   execution time : 24.581 s
Press any key to continue.
```




```
Enter number of elements in array: 4

Enter 4 numbers
3
5
7
1

Enter a number to search: 2
2 isn't present in the array.

Process returned 0 (0x0)   execution time : 13.586 s
Press any key to continue.
```

Experiment No: 4 **Date:** 11 – 02 - 2023

Experiment Name: Write a program to search specific data from a list using binary search.

Code:

```
// Binary Search
#include <stdio.h>
int main()
{
   int c, first, last, middle, n, search, array[100];
   printf("Enter the number of elements: ");
   scanf("%d", &n);
   if (n<=0)
      printf("Invalid!");
   else
   {
      printf("\nEnter %d integers in ascending order\n", n);
      for (c = 0; c < n; c++)
         scanf("%d",  &array[c]);
      printf("\nEnter value to find in the list: ");
      scanf("%d", &search);
      first = 0;
      last = n - 1;
      middle = (first+last)/2;
      while (first <= last)
      {
         if (array[middle] < search)
            first = middle + 1;
         else if (array[middle] == search)
         {
            printf("%d found at location %d\n", search, middle+1);
            break;
         }
```

```
        else
            last = middle - 1;
        middle = (first + last)/2;
      }
      if (first > last)
        printf("Not found!\n%d isn't present in the list\n", search);
   }
}
```

Output:

```
"C:\Users\USER\Desktop\dsa cd\4.exe"
Enter the number of elements: -5
Invalid!
Process returned 0 (0x0)   execution time : 2.541 s
Press any key to continue.
"C:\Users\USER\Desktop\dsa cd\4.exe"
Enter the number of elements: 5

Enter 5 integers in ascending order
-5
-3
-1
3
5

Enter value to find in the list: 5
5 found at location 5

Process returned 0 (0x0)   execution time : 21.050 s
Press any key to continue.
"C:\Users\USER\Desktop\dsa cd\4.exe"
Enter the number of elements: 4

Enter 4 integers in ascending order
-3
-2
0
8

Enter value to find in the list: 1
Not found!
1 isn't present in the list

Process returned 0 (0x0)   execution time : 12.681 s
Press any key to continue.
```

Experiment No: 5 **Date:** 18 - 02- 2023

Experiment Name: Write a program to perform some stack operation (push, pop, print) using array.

Code:

```
// Perform some stack operation (push, pop, print) using array
#include<stdio.h>
int stack[100],choice,n,top,x,i;
void push(void);
void pop(void);
void display(void);
int main( )
{
   top=-1;
   printf("Enter the STACK size (maximum 100):");
   scanf("%d",&n);
   printf("\nEnter what operations you want to perform :");
   printf("\n\t 1.PUSH\n\t 2.POP\n\t 3.DISPLAY\n\t 4.EXIT\n");
   do
   {
      printf("\nEnter the Choice: ");
      scanf("%d",&choice);
      switch(choice)
      {
      case 1:
      {
         push( );
         break;
      }
      case 2:
      {
         pop( );
```

```
            break;
        }
        case 3:
        {
            display( );
            break;
        }
        case 4:
        {
            printf("\n\t EXIT POINT ");
            break;
        }
        default:
        {
            printf ("\n\tPlease Enter a Valid Choice between 1/2/3/4");
        }
        }
    }
    while(choice!=4);
    return 0;
}
void push( )
{
    if(top>=n-1)
    {
        printf("\n\tSTACK is FULL!");
    }
    else
    {
        printf("Enter a value you want to push in stack: ");
        scanf("%d",&x);
```

```
        top++;
        stack[top]=x;
    }
}
void pop( )
{
    if(top<=-1)
    {
        printf("\n\t Stack is EMPTY!");
    }
    else
    {
        printf("\nThe popped element is %d",stack[top]);
        top--;
    }
}
void display( )
{
    if(top>=0)
    {
        printf("\nThe elements in STACK are :\n");
        for(i=top; i>=0; i--)
            printf("\n%d",stack[i]);
    }
    else
    {
        printf("\nThe STACK is EMPTY!");
    }
}
```

Output:

```
"C:\Users\USER\Desktop\dsa cd\5.exe"
Enter the STACK size (maximum 100):20

Enter what operations you want to perform :
         1.PUSH
         2.POP
         3.DISPLAY
         4.EXIT

Enter the Choice: 3

The STACK is EMPTY!
Enter the Choice: 1
Enter a value you want to push in stack: 23

Enter the Choice: 1
Enter a value you want to push in stack: 32

Enter the Choice: 1
Enter a value you want to push in stack: 45

Enter the Choice: 1
Enter a value you want to push in stack: 54

Enter the Choice: 3

The elements in STACK are :

54
45
32
23
Enter the Choice: 2

The popped element is 54
Enter the Choice: 3

The elements in STACK are :

45
32
23
Enter the Choice: 4

         EXIT POINT
Process returned 0 (0x0)   execution time : 39.496 s
Press any key to continue.
```

Experiment No: 6 **Date:** 25 - 02 - 2022

Experiment Name: Write a program to perform some queue operation (enqueue, dequeue, display) using array.

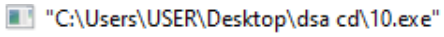
Code:

```
// Perform some queue operation (enqueue, dequeue, display) using array
#include <stdio.h>
#define SIZE 20
void enqueue();
void dequeue();
void display();
int items[SIZE], front = -1, rear = -1,choice;
int main( )
{
   printf("Choose the option you want to be performed  :");
   printf("\n\t 1.ENQUEUE\n\t 2.DEQUEUE\n\t 3.DISPLAY\n\t 4.EXIT");
   printf("\n");
   do
   {
      printf("\nEnter the Choice:");
      scanf("%d",&choice);
      switch(choice)
      {
      case 1:
      {
         enqueue( );
         break;
      }
      case 2:
      {
         dequeue( );
         break;
```

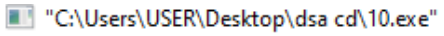
```
        }
        case 3:
        {
            display( );
            break;
        }
        case 4:
        {
            printf("\n\t EXIT POINT ");
            break;
        }
        default:
        {
            printf ("\n\tPlease Enter a Valid Choice between 1/2/3/4");
        }
        }
    }
    while(choice!=4);
    return 0;
}
void enqueue()
{
    int item;
    if(rear == SIZE - 1)
        printf("Queue FULL! \n");
    else
    {
        if(front== - 1)
            front = 0;
        printf("Inset the element in queue : ");
        scanf("%d", &item);
```

```
    {
        struct node* temp = head;
        while(temp->next->next!=NULL)
        {
            temp = temp->next;
        }
        temp->next = NULL;
    }
    else
    {
        struct node* temp = head;
        for(int i=2; i< f; i++)
        {
            if(temp->next!=NULL)
            {
                temp = temp->next;
            }
        }
        temp->next = temp->next->next;
    }
    printf("\nAfter deleting an element, the remaining elements are: ");
    printLinkedlist(head);
}
```

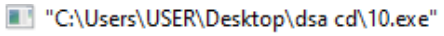
Output:



```
"C:\Users\USER\Desktop\dsa cd\10.exe"
Enter 1st value : 2
Enter 2nd value : 4
Enter 3rd value : 6
Enter 4th value : 8
Enter 5th value : 9
Enter 6th value : 7

Enter the position from 1-6, which element you want to delete: 1

After deleting an element, the remaining elements are: 4 6 8 9 7
Process returned 0 (0x0)   execution time : 25.779 s
Press any key to continue.
```



```
"C:\Users\USER\Desktop\dsa cd\10.exe"
Enter 1st value : 2
Enter 2nd value : 4
Enter 3rd value : 6
Enter 4th value : 8
Enter 5th value : 9
Enter 6th value : 7

Enter the position from 1-6, which element you want to delete: 4

After deleting an element, the remaining elements are: 2 4 6 9 7
Process returned 0 (0x0)   execution time : 10.813 s
Press any key to continue.
```



```
"C:\Users\USER\Desktop\dsa cd\10.exe"
Enter 1st value : 2
Enter 2nd value : 4
Enter 3rd value : 6
Enter 4th value : 8
Enter 5th value : 9
Enter 6th value : 7

Enter the position from 1-6, which element you want to delete: 6

After deleting an element, the remaining elements are: 2 4 6 8 9
Process returned 0 (0x0)   execution time : 7.690 s
Press any key to continue.
```

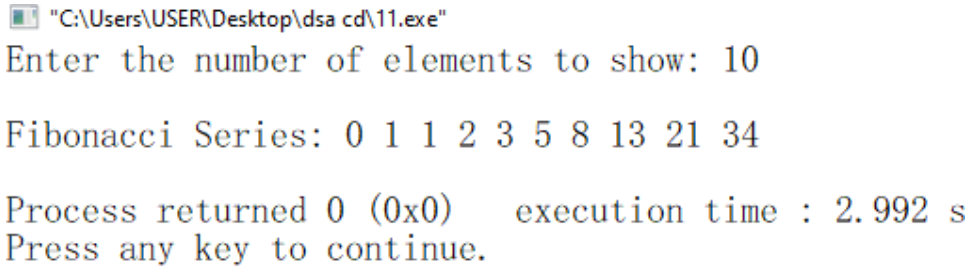
Experiment No: 7 **Date:** 04 - 03 - 2023

Experiment Name: Write a program to print Fibonacci series using recursion.

Code:

```
// Fibonaci Series
#include<stdio.h>
void Fibonacci(int num)
{
   static int num1=0,num2=1,num3;
   if(num>0)
   {
      num3 = num1 + num2;
      num1 = num2;
      num2 = num3;
      printf("%d ",num3);
      Fibonacci(num-1);
   }
}
int main()
{
   int num;
   printf("Enter the number of elements to show: ");
   scanf("%d",&num);
   printf("\nFibonacci Series: ");
   printf("%d %d ",0,1);
   Fibonacci(num-2);
   printf("\n");
   return 0;
}
```

Output:



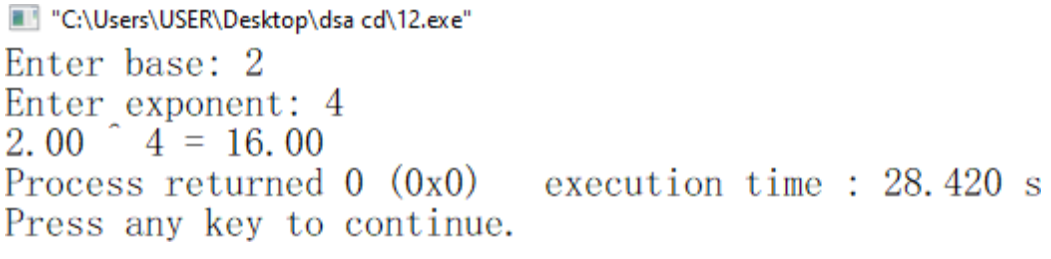
Experiment No: 08 **Date:** 04 - 03 - 2023

Experiment Name: Write a program to calculate power using recursion.

Code:

```
// Calculate power using recursion
#include <stdio.h>
double pow(double base, double exponent);
int main()
{
    double base, power;
    int exponent;
    printf("Enter base: ");
    scanf("%lf", &base);
    printf("Enter exponent: ");
    scanf("%d", &exponent);
    power = pow(base, exponent);
    printf("%.2lf ^ %d = %.2lf", base, exponent, power);
    return 0;
}
double pow(double base, double exponent)
{
    if(exponent == 0)
        return 1;
    else if(exponent > 0)
        return base * pow(base, exponent - 1);
    else
        return 1 / pow(base, -exponent);
}
```

Output:



Experiment No: 09 **Date:** 18 - 03 - 2023

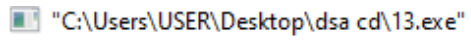
Experiment Name: Write a program to calculate permutation using recursion.

Code:

```
// Calculate permutation using recursion
#include <stdio.h>
#include <string.h>
int count=0;
void swap(char *x, char *y)
{
    char temp;
    temp = *x;
    *x = *y;
    *y = temp;
}
void permutation(char s[], int l, int r)
{
    if (l == r)
    {
        puts(s);
        count++;
    }
    else
    {
        for (int i = l; i <= r; i++)
        {
            swap(&s[l], &s[i]);
            permutation(s, l+1, r);
            swap(&s[l], &s[i]);
        }
    }
}
```

```
int main()
{
   char str[100];
   printf("Enter Expression: ");
   gets(str);
   int n = strlen(str);
   permutation(str, 0, n-1);
   printf("\nTotal Permutation:%d\n",count);
   return 0;
}
```

Output:




```
Enter Expression: love
love
loev
lvoe
lveo
levo
leov
olve
olev
ovle
ovel
oevl
oelv
vole
voel
vloe
vleo
velo
veol
eovl
eolv
evol
evlo
elvo
elov

Total Permutation:24

Process returned 0 (0x0)   execution time : 2.350 s
Press any key to continue.
```

Experiment No: 10 **Date:** 18 - 03 - 2023

Experiment Name: Write a program to calculate combination using recursion.

Code:

```
// Calculate combination using recursion
#include <stdio.h>
int NCR (int n, int r)
{
   if (r == 0 || n == r)
   {
      return 1;
   }
   else
      return NCR (n - 1, r - 1) + NCR (n - 1, r);
}
int main ()
{
   int n,r;
   printf("Enter a number n: ");
   scanf("%d",&n);
   printf("Enter a number r: ");
   scanf("%d",&r);
   printf("\nValue of %dC%d = %d\n",n,r, NCR (n, r));
}
```

Output: