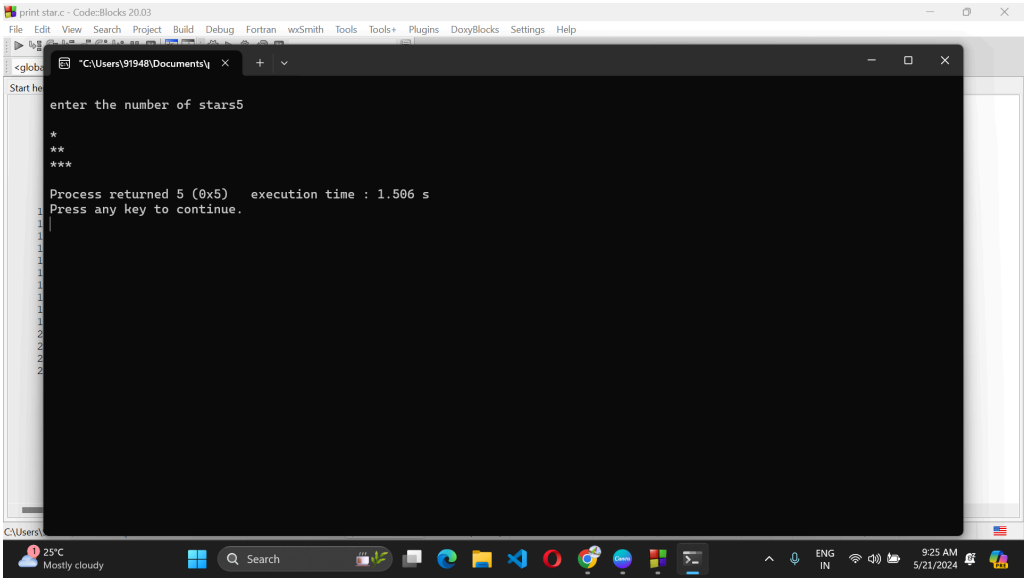
program 1: print the stars

```
#include <stdio.h>

#include <conio.h>



void main ()

{



   int i,j,ns=0;

   system("cls");

   printf("\nenter the number of stars");

   scanf("%d",&ns);

   for( i=1;i<ns;i++)

   {



      for(j=2;j<=i;j++)

      {



         printf("*");



      }

      printf("\n");

   }

}
```

output:

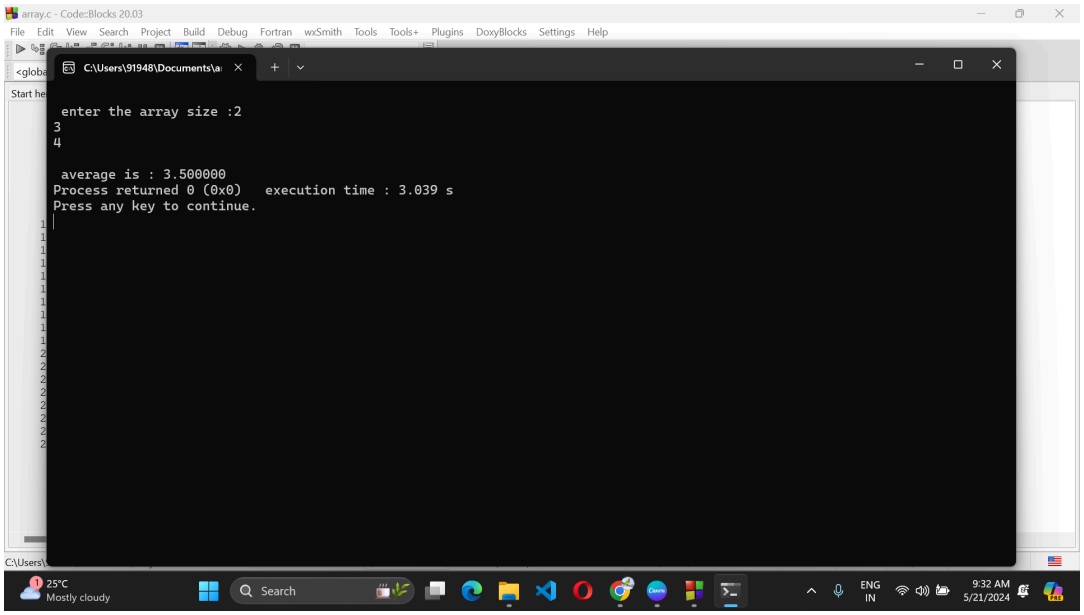


program 2: Array

```
#include <stdio.h>

float avg( int arr[], int size);

main()

 {

    int x[100],k,n;

   printf("\n enter the array size :" );

   scanf("%d",&n);



   for(k=0;k<n;k++)

   {

      scanf("%d",&x[k]);

      }

      printf("\n average is : %f",avg(x,n));

      }

float avg( int arr[], int size)

{

  int *p,i,sum=0;

  p=arr;
```

```
 for(int i=0;i<size;i++)
 {
    sum=sum+ *(p+i);


 }
 return (float) sum/size;
}
```

output :



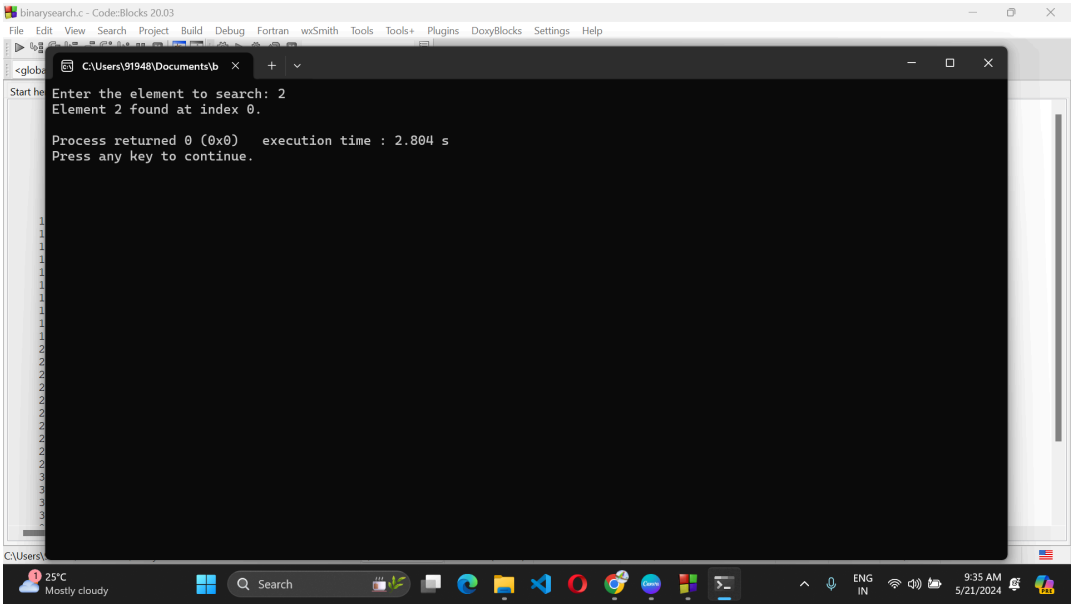
program 3: Binary search

```
#include <stdio.h>


int binarySearch(int arr[], int n, int key) {
   int low = 0;
   int high = n - 1;


   while (low <= high) {
      int mid = low + (high - low) / 2;
```

```
        if (arr[mid] == key) {
            return mid;
        } else if (arr[mid] < key) {
            low = mid + 1;
        } else {
            high = mid - 1; // Search in the left half
        }
    }

    return -1; // Element not found
}

int main() {
    int arr[] = {2, 5, 8, 12, 16, 23, 38, 56, 72, 91};
    int n = sizeof(arr) / sizeof(arr[0]);
    int key;

    printf("Enter the element to search: ");
    scanf("%d", &key);

    int index = binarySearch(arr, n, key);

    if (index != -1) {
        printf("Element %d found at index %d.\n", key, index);
    } else {
        printf("Element %d not found in the array.\n", key);
    }
```

```
    return 0;
}
```

output



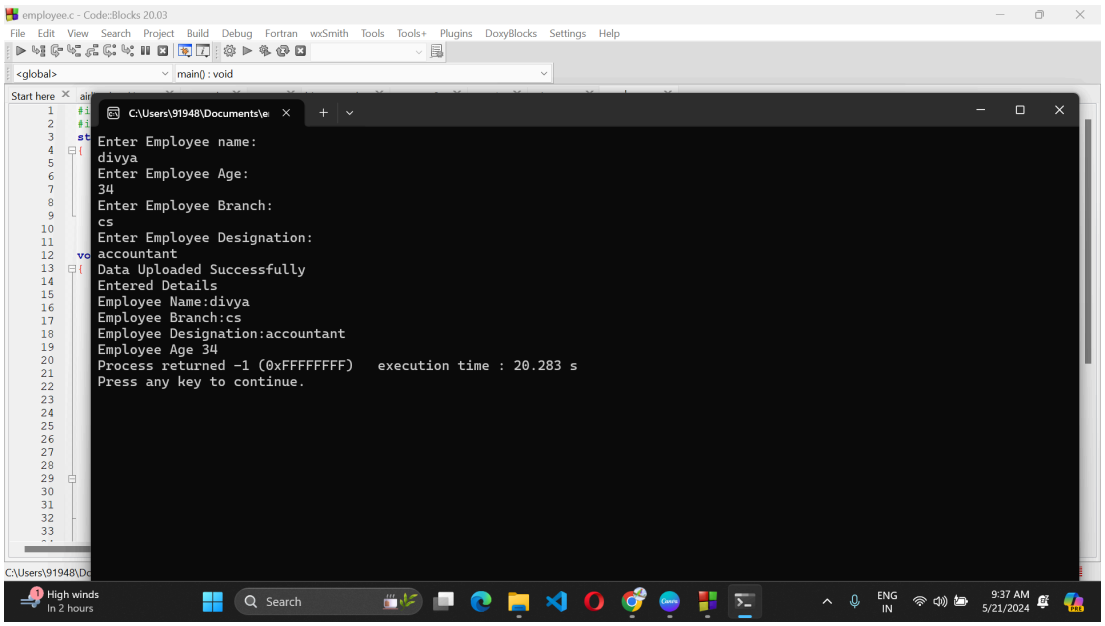
program 4: Employee list

```
#include <stdio.h>
#include <string.h>
struct emp
{
   char name[30];
   int age;
   char branch[20];
   char designation[20];
   };



void main()
{
```

```
   struct emp p1;
   printf("Enter Employee name: \n");
   scanf("%s", p1.name);
   strcpy (p1.name, p1.name);

   printf("Enter Employee Age: \n");
   scanf("%d", &p1.age);

   printf("Enter Employee Branch:\n");
   scanf("%s", p1.branch);

   printf("Enter Employee Designation:\n");
   scanf("%s", p1.designation);

   FILE *fp=fopen("c:\\Assignments\\Demo.txt", "a");
   if(fp==NULL){
      printf("Error opening File\n");
      return ;
   }
   fprintf(fp,"Employee Name:%s\n Employee Age: %d \n Employee Branch:%s\n Employee Designation :%s\n
",p1.name,p1.age,p1.branch,p1.designation);
   printf("Data Uploaded Successfully\n");

   printf("Entered Details\n");
   printf("Employee Name:%s\n",p1.name);
   printf("Employee Branch:%s\n",p1.branch);
   printf("Employee Designation:%s\n",p1.designation);
```

```
    printf("Employee Age %d",p1.age);


    FILE *ptr;
    ptr=fopen("C:\\Assignments\\Demo.txt", "a");
    if(ptr==NULL){
        printf("Failed to open File\n");
        return;
    }
    char a[100];
    while(fscanf(ptr,"%s",a)!=EOF){
        printf("%s\n",a);
    }


}
```

output :



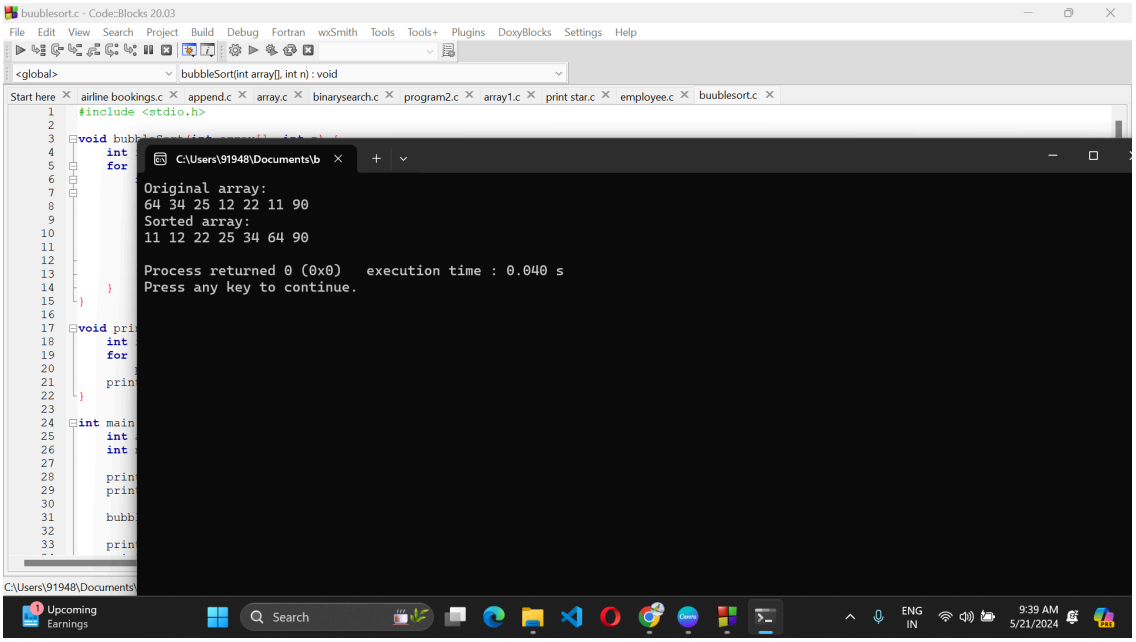
program 5 : Bubble sort

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

```
void bubbleSort(int array[], int n) {
   int i, j, temp;
   for (i = 0; i < n-1; i++) {
      for (j = 0; j < n-i-1; j++) {
         if (array[j] > array[j+1]) {

            temp = array[j];
            array[j] = array[j+1];
            array[j+1] = temp;
         }
      }
   }
}

void printArray(int array[], int size) {
   int i;
   for (i=0; i < size; i++)
      printf("%d ", array[i]);
   printf("\n");
}

int main() {
   int array[] = {64, 34, 25, 12, 22, 11, 90};
   int n = sizeof(array)/sizeof(array[0]);

   printf("Original array: \n");
   printArray(array, n);
```

```
    bubbleSort(array, n);

    printf("Sorted array: \n");
    printArray(array, n);

    return 0;
}
```

output

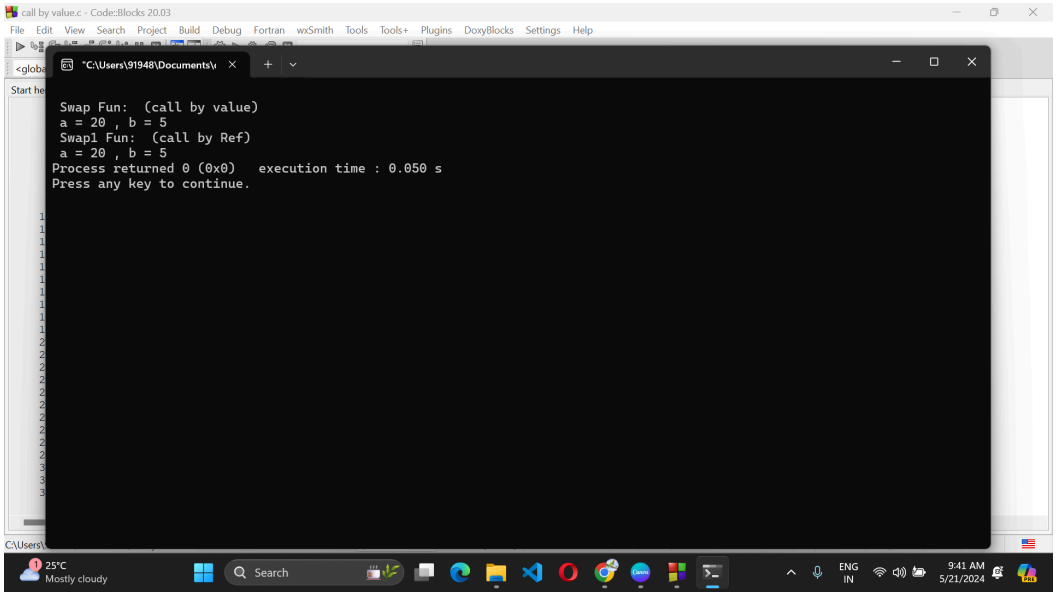


program 6: Call by value

```
#include <stdio.h>


main()
{
    int a,b;
    a=5, b=20;
    swap (a,b);
    swap1 (&a, &b);
```

```
   printf ("\n Swap Fun:  (call by value)  \n a = %d , b = %d ", a,b);
   printf ("\n Swap1 Fun:  (call by Ref)  \n a = %d , b = %d ", a,b);

}

void swap (int x, int y)
{

   int tmp;
   tmp = x;
   x=y;
   y=tmp;

}
void swap1 (int *x1, int *y1)
{

   int tmp1;
   tmp1 = *x1;
   *x1=*y1;
   *y1=tmp1;

}
```

output



program 7: Calender

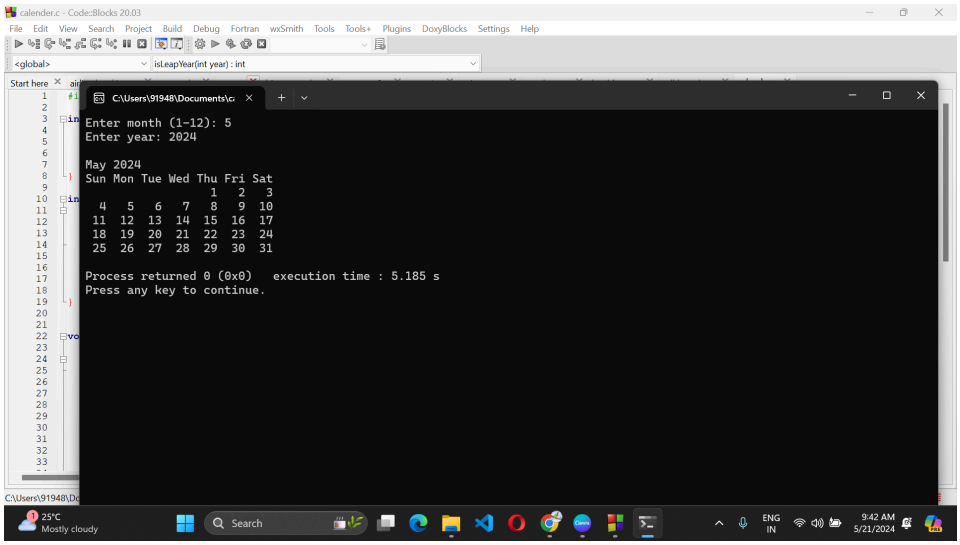
```
#include <stdio.h>


int isLeapYear(int year) {
   if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0))
      return 1;
   else
      return 0;
}


int getDayOfWeek(int day, int month, int year) {
   if (month < 3) {
      month += 12;
      year -= 1;
   }
   int K = year % 100;
   int J = year / 100;
   int h = (day + ((13 * (month + 1)) / 5) + K + (K / 4) + (J / 4) + (5 * J)) % 7;
   return h;
```

```
}



void printCalendar(int month, int year) {
    int daysInMonth[] = {31, 28 + isLeapYear(year), 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
    char* monthNames[] = {"January", "February", "March", "April", "May", "June", "July",
                          "August", "September", "October", "November", "December"};



    printf("\n%s %d\n", monthNames[month - 1], year);
    printf("Sun Mon Tue Wed Thu Fri Sat\n");



    int dayOfWeek = getDayOfWeek(1, month, year);



    for (int i = 0; i < dayOfWeek; i++) {
        printf("    ");
    }



    for (int day = 1; day <= daysInMonth[month - 1]; day++) {
        printf("%3d ", day);
        if (++dayOfWeek > 6) {
            printf("\n");
            dayOfWeek = 0;
        }
```

```
    }

    if (dayOfWeek != 0) {
        printf("\n");
    }
}

int main() {
    int month, year;

    printf("Enter month (1-12): ");
    scanf("%d", &month);
    printf("Enter year: ");
    scanf("%d", &year);

    if (month < 1 || month > 12 || year < 0) {
        printf("Invalid input!\n");
        return 1;
    }

    printCalendar(month, year);

    return 0;
```

}

output:

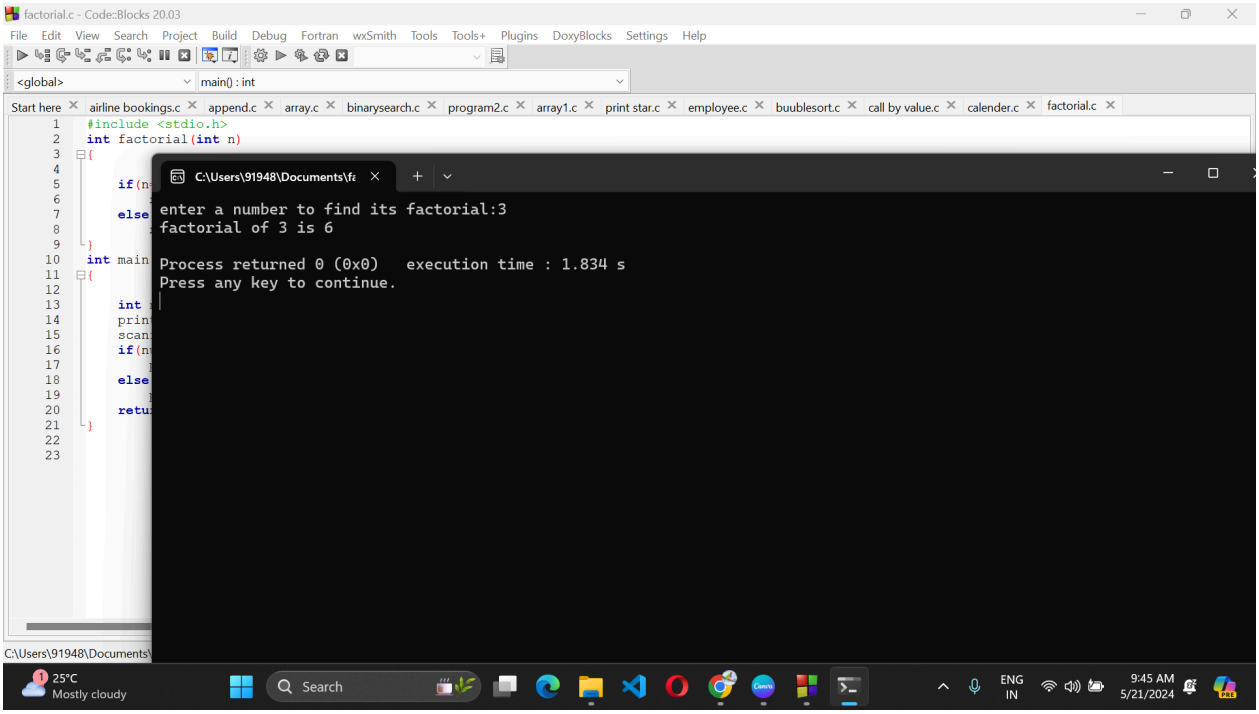


program 8 : Factoral

```
#include <stdio.h>

int factorial(int n)

{


   if(n==0)

      return 1;

   else

      return n*factorial(n -1);

}

int main()

{


   int num;

   printf("enter a number to find its factorial:");

   scanf("%d",&num);

   if(num < 0)
```

```
    printf("factorial is not defined for negative number\n");
  else
    printf("factorial of %d is %d\n",num,factorial(num));
  return 0;
}
```

output:

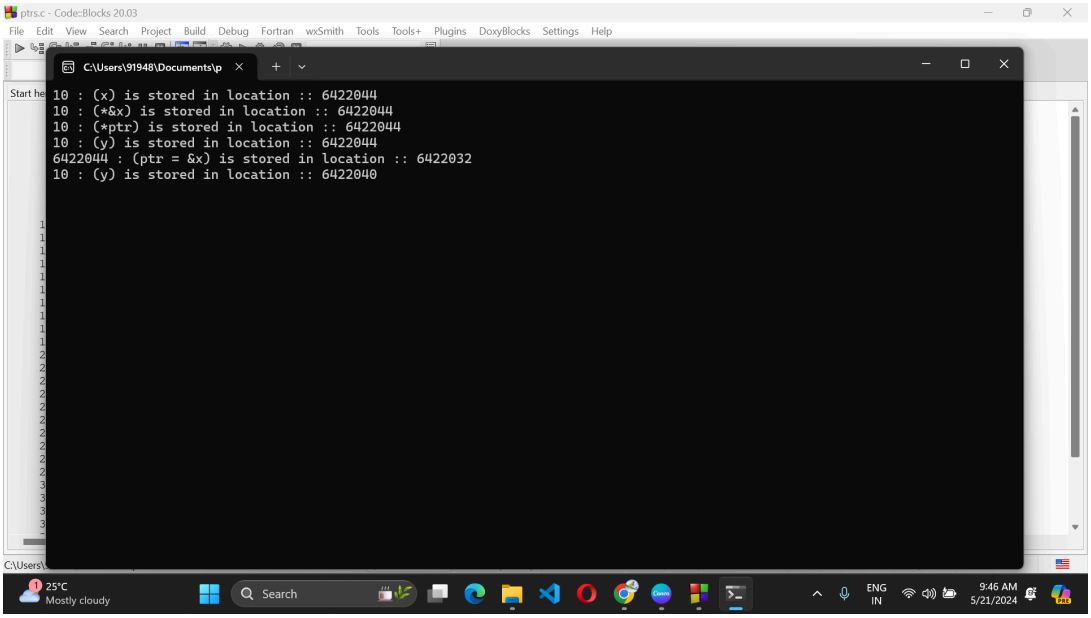


program 9 : pointers

```
#include <stdio.h>
main()
{
  /*int 1;
  float b,c;
  double d;
  char c;
  a=5;b=3.7;c=10.80;d=12345678.99;ch='s';
  printf("%d is stored in the location :: %u \n",a,&a);*/
  /*int *p,n;
```

```
    p=&n;
    n=0x18;
    *p=*p+4;
    printf("%u is stored in the location :: %u \n",n,&n);
    printf("%u is stored in the location :: %u \n",p,&p);
    getch();
    */
    /*&x, &x,
*ptr, ptr
y, &*ptr
ptr, &ptr
y, &y*/
    int x,y;
   int *ptr;
   x=10;
   ptr = &x;
   y= *ptr;
   printf ("%d : (x) is stored in location :: %u \n", x, &x);
   printf ("%d : (*&x) is stored in location :: %u \n", *&x, &x);
   printf ("%d : (*ptr) is stored in location :: %u \n", *ptr, ptr);
   printf ("%d : (y) is stored in location :: %u \n", y, &*ptr);
   printf ("%u : (ptr = &x) is stored in location :: %u \n", ptr, &ptr);
   printf ("%d : (y) is stored in location :: %u \n", y, &y);
getch();
 }
```

output:



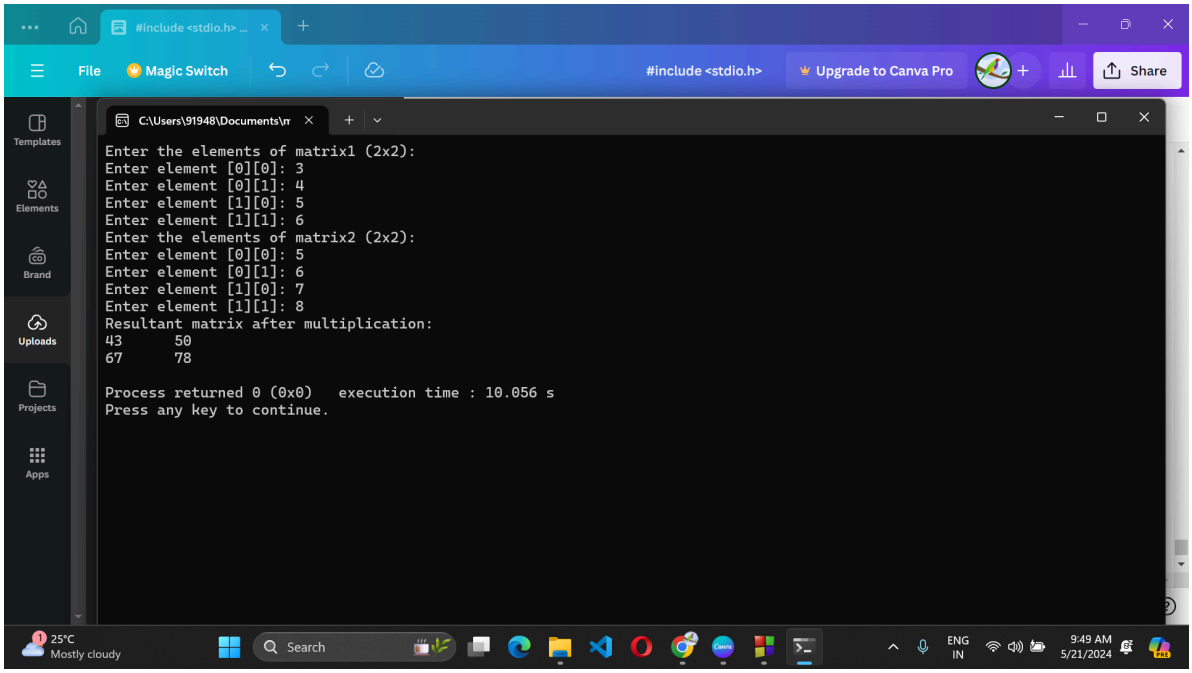
program 10 : Matrix

```
#include <stdio.h>

#include <conio.h>


int main() {

   int matrix1[2][2];

   int matrix2[2][2];

   int result[2][2];

   int i, j, k;



   printf("Enter the elements of matrix1 (2x2):\n");

   for (i = 0; i < 2; i++) {

      for (j = 0; j < 2; j++) {

         printf("Enter element [%d][%d]: ", i, j);

         scanf("%d", &matrix1[i][j]);

      }
```

```
}

printf("Enter the elements of matrix2 (2x2):\n");
for (i = 0; i < 2; i++) {
   for (j = 0; j < 2; j++) {
      printf("Enter element [%d][%d]: ", i, j);
      scanf("%d", &matrix2[i][j]);
   }
}

for (i = 0; i < 2; i++) {
   for (j = 0; j < 2; j++) {
      result[i][j] = 0;
      for (k = 0; k < 2; k++) {
         result[i][j] += matrix1[i][k] * matrix2[k][j];
      }
   }
}

printf("Resultant matrix after multiplication:\n");
for (i = 0; i < 2; i++) {
   for (j = 0; j < 2; j++) {
      printf("%d\t", result[i][j]);
   }
   printf("\n");
}
```

```
    return 0;
}
```

output



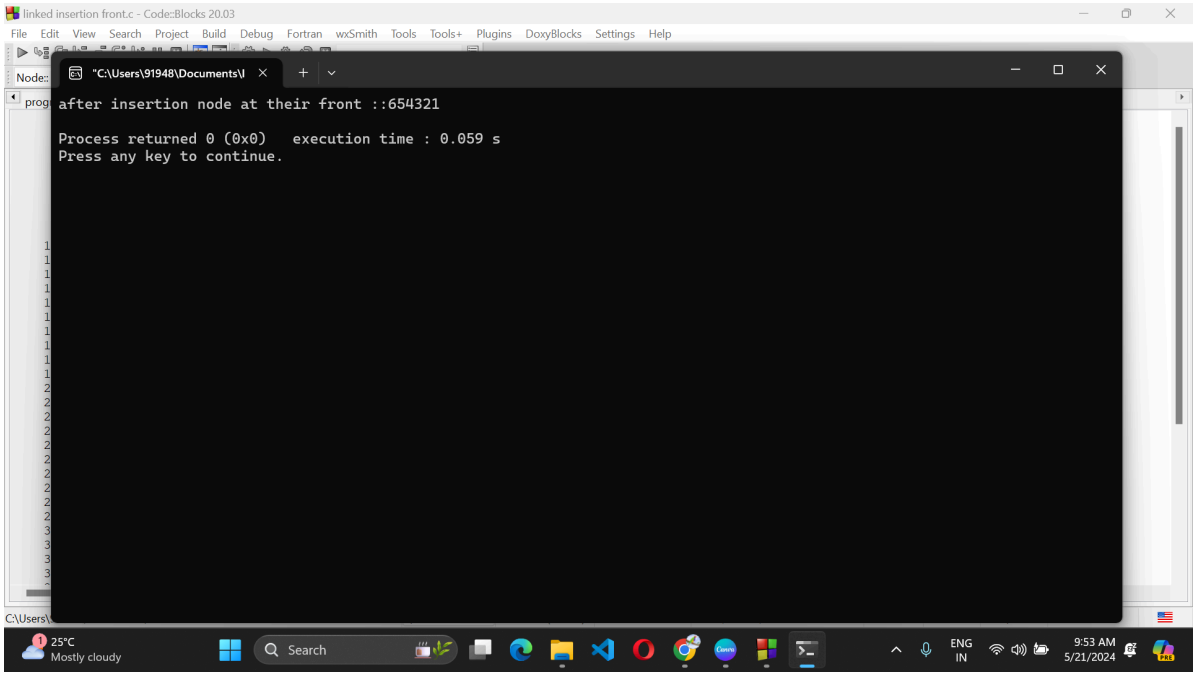
program 11 : Linked list insertion from front

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
 struct Node
 {

    int data;
    struct Node* next;
 };

 void insertAtFront(struct Node**head_ref,int new_data)
 {
```

```
    struct Node* new_node=(struct Node*)malloc(sizeof(struct Node));



    new_node->data = new_data;
    new_node->next = (*head_ref);
    (*head_ref)=new_node;
}



    void printList(struct Node* node)
    {
        while(node!=NULL)
        {

            printf("%d",node->data);
            node=node->next;
        }
            printf("\n");

        }
        int main()
              {
                  struct Node* head=NULL;

                  insertAtFront(&head,1);
                   insertAtFront(&head,2);
                    insertAtFront(&head,3);
```

```
        insertAtFront(&head,4);
        insertAtFront(&head,5);
        insertAtFront(&head,6);
        printf("after insertion node at their front ::");
        printList(head);
    return 0;

    }
```

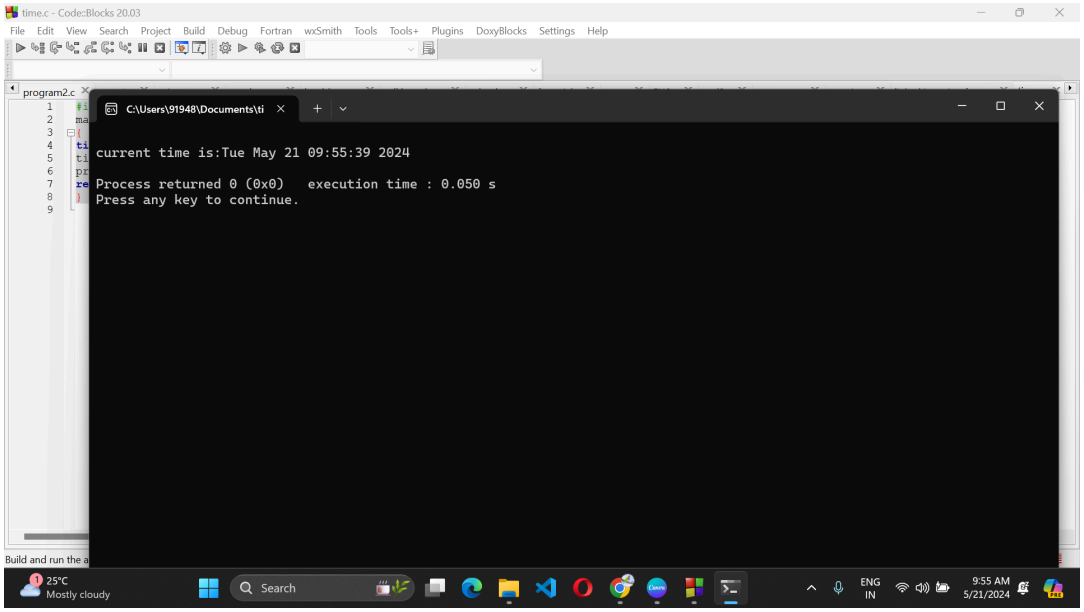
output :



program 12 : Current time

```
#include <stdio.h>

main()

{

time_t t;

time(&t);

printf("\ncurrent time is:%\s",ctime(&t));

return 0;

}
```

output :



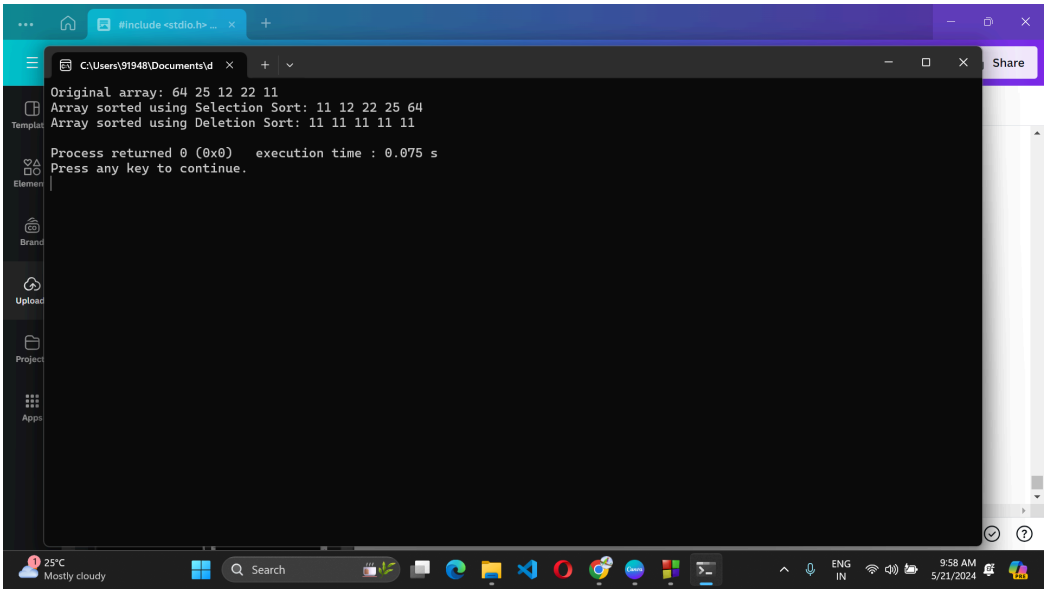
program 13 : Arrays

```
#include <stdio.h>



void selectionSort(int arr[], int n) {
   int i, j, min_index;

   for (i = 0; i < n-1; i++) {
      min_index = i;
      for (j = i+1; j < n; j++) {
         if (arr[j] < arr[min_index]) {
            min_index = j;
         }
      }

      int temp = arr[min_index];
      arr[min_index] = arr[i];
      arr[i] = temp;
```

```
    }
}



void deletionSort(int arr[], int n) {
    int i, j;

    for (i = 0; i < n-1; i++) {
        for (j = i+1; j < n; j++) {
            if (arr[j] < arr[i]) {

                int k;
                for (k = i; k < n-1; k++) {
                    arr[k] = arr[k+1];
                }
                n--;
                i--;
                break;
            }
        }
    }
}

void printArray(int arr[], int n) {
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
```

```
    printf("\n");
}

int main() {
    int arr[] = {64, 25, 12, 22, 11};
    int n = sizeof(arr)/sizeof(arr[0]);

    printf("Original array: ");
    printArray(arr, n);

    selectionSort(arr, n);
    printf("Array sorted using Selection Sort: ");
    printArray(arr, n);

    int arr2[] = {64, 25, 12, 22, 11};
    int n2 = sizeof(arr2)/sizeof(arr2[0]);

    deletionSort(arr2, n2);
    printf("Array sorted using Deletion Sort: ");
    printArray(arr2, n2);

    return 0;
}
```

output :



program 14 : Conversions

```
#include<stdio.h>

#include<string.h>

#include<math.h>

void start();

void convert();

void binary_Values();

int bin_to_dec();

void bin_to_oct();

void bin_to_hexa();


void decimal_values();

void dec_to_binary();

void dec_to_oct();

void dec_to_hexa();


void octal_values();

void oct_bin();
```

```
void hexa_values();

void hexa_bin();


void wish();


void wish(){
   wis:
   printf("---------------------------------\n");
   printf("Do you wish to Continue?\n");
   printf("Press Y to continue or press N to cancel.\n");
   printf("---------------------------------\n");
   char c;
   scanf("%c", &c);
   if (c == 'Y' || c == 'y')
   {
      system("cls");
      start();
   }
   else if(c=='N' || c == 'n') {
      printf("Exiting...\n");
      return ;
   }
   else{
      printf("Please enter a valid Keyword.\n");
      goto wis;
   }
```

```
}
void start(){
   start:
   printf("press 1 for Binary Conversions\n");
   printf("Press 2 for Decimal Conversions\n");
   printf("Press 3 for Octal Conversions\n");
   printf("Press 4 for HexDecimal Conversions\n");
   printf("Press 5 for Exit Conversions\n");
   printf("---------------------------------\n");
   int option;
   scanf("%d", &option);
   if(option ==5){
      printf("Exiting.....!\n");
      return ;
   }
   else if(option>4){
      printf("Choose the correct option\n");
      printf("---------------------------------\n");
      goto start;
   }
   else{
      convert(option);
   }
}
void convert(int a){
   switch (a)
   {
```

```
    case 1:
        binary_Values();
        wish();
        break;
    case 2:
        decimal_values();
        wish();
        break;
    case 3:
        octal_values();
        wish();
        break;
    case 4:
        hexa_values();
        wish();
        break;
    default:
        break;
    }
}
void binary_Values(){
    bin:
    printf("Please Enter the Binary Value\n");
    char a[20];
    scanf("%s",a);
    for(int i=0;i<20;i++)
    {
```

```
        if(!a[i]==0 && !a[i]==1){
            printf("Enter the Binary Values only\n");
            goto bin;
        }
    }
    printf("Decimal value of %s is : %d\n",a,bin_to_dec(a));
    bin_to_oct(a);
    bin_to_hexa(a);

}
int bin_to_dec(char a[]){
    int dec=0;
    int c=0;
    for(int i=strlen(a)-1;i>=0;i--){
        dec=dec+(a[i]-'0')*(int)pow(2,c);
        c++;
    }
    return dec;
}
void bin_to_oct(char a[]){
    int dec=bin_to_dec(a);
    char a1[20];
    sprintf(a1,"%o",dec);
    printf("Octal value of %s is : %s\n",a,a1);
}
void bin_to_hexa(char a[]){
    int dec=bin_to_dec(a);
```

```
    char a1[20];
    sprintf(a1,"%x",dec);
    printf("Hexa value of %s is: %s\n",a,a1);
}

void decimal_values(){
    printf("Enter a decimal value:\n");
    int i;
    scanf("%d",&i);
    if(!i>0){
        printf("Enter a decimal value\n");
    }
    dec_to_binary(i);
    dec_to_oct(i);
    dec_to_hexa(i);
}
void dec_to_binary(int a){
    int c=0;
    int bin[20];
    while(a>0){
        bin[c++]=a%2;
        a=a/2;
    }
    printf("Binary value of %d is :",a);
    for(int i=c-1;i>=0;i--){

        printf("%d",bin[i]);
```

```
    }
    printf("\n");
}
void dec_to_oct(int a){
    char dec[20];
    sprintf(dec, "%o",a);
    printf("Octal value of %d is :%s\n",a,dec);
}
void dec_to_hexa(int a){
    char hex[20];
    sprintf(hex, "%x",a);
    printf("Hexadecimal value of %d is :%s\n",a,hex);
}
void octal_values(){
    printf("Enter your Octal Number:\n");
    char i[20];
    scanf("%s",i);
    oct_bin(i);
}
void oct_bin(char a[]){
    int dec=0;
    int c=0;
    for(int i=strlen(a)-1;i>=0;i--){
        dec=dec+(a[i]-'0')*(int)pow(8,c);
        c++;
    }
    printf("Decimal Value of %s is :%d\n",a,dec);
```

```
    dec_to_binary(dec);
    dec_to_hexa(dec);
}

void hexa_values(){
     hexa:
    printf("Enter the hexadecimal Values\n");
    char c[20];
    scanf("%s",c);
    for(int i=0;i<strlen(c);i++){
          if((c[i]>='0' && c[i]<='9') || (c[i]>='a' && c[i]<='f')){
          }
          else{
             printf("Invalid hexadecimal value\n");
             // system("cls");
             goto hexa;
          }
    }
    hexa_bin(c);
}
void hexa_bin(char a[]){
    int dec=0;
    int c=0;
    for(int i=strlen(a)-1;i>=0;i--){
       if((a[i]>='0' && a[i]<='9')){
          dec=dec+(a[i]-'0')*(int)pow(16,c);
       }
```

```
    else{
        dec=dec+(a[i]-'a'+10)*(int)pow(16,c);
    }
    c++;
  }
  printf("Decimal Value of %s is: %d\n",a,dec);
  dec_to_binary(dec);
  dec_to_oct(dec);
}
int main(){
  printf("Welcome to OMG Conersions!\n");
  start();
}
```

output :