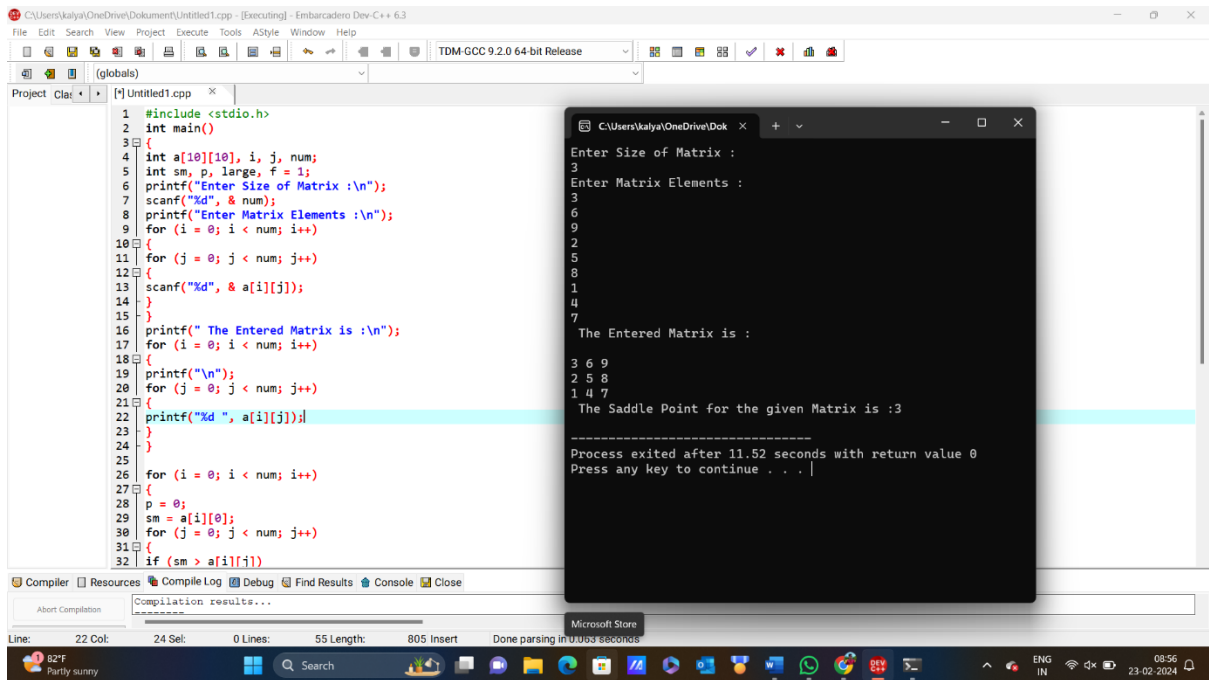


C programming

Day – 3 class work:

1. Find the saddle point of a given matrix.

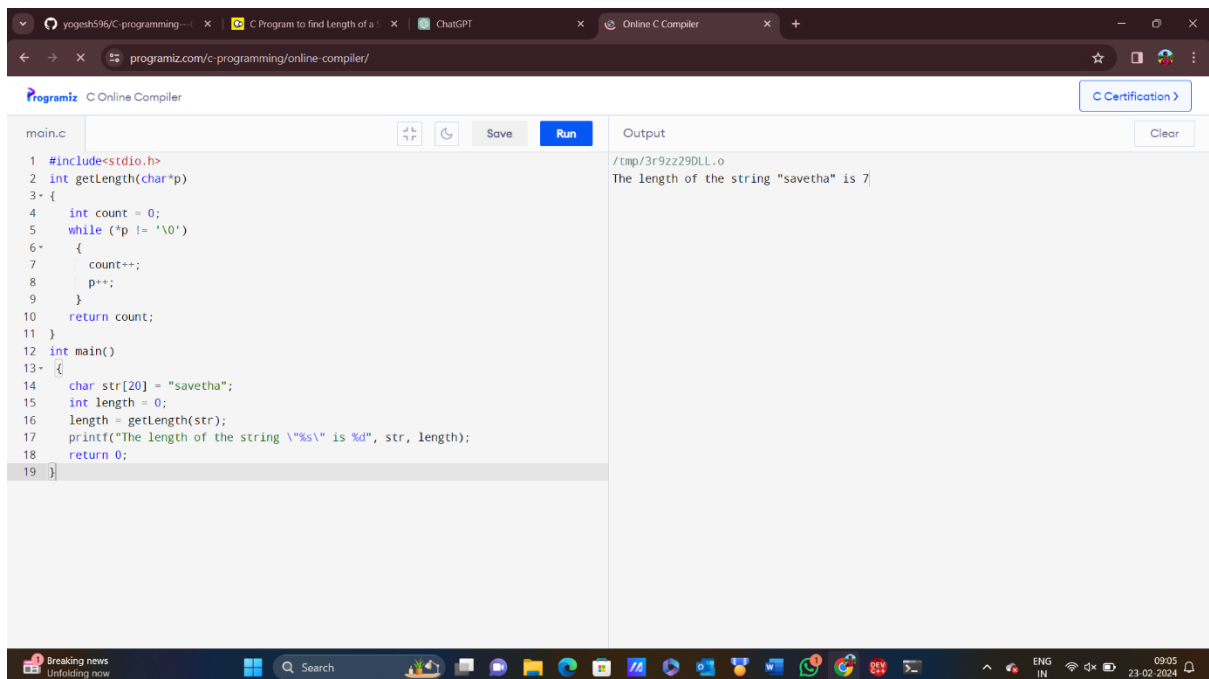


```
1 #include <stdio.h>
2 int main()
3 {
4     int a[10][10], i, j, num;
5     int sm, p, large, f = 1;
6     printf("Enter Size of Matrix :\n");
7     scanf("%d", &num);
8     printf("Enter Matrix Elements :\n");
9     for (i = 0; i < num; i++)
10     {
11         for (j = 0; j < num; j++)
12         {
13             scanf("%d", &a[i][j]);
14         }
15     }
16     printf(" The Entered Matrix is :\n");
17     for (i = 0; i < num; i++)
18     {
19         printf("\n");
20         for (j = 0; j < num; j++)
21         {
22             printf("%d ", a[i][j]);
23         }
24     }
25     for (i = 0; i < num; i++)
26     {
27         p = 0;
28         sm = a[i][0];
29         for (j = 0; j < num; j++)
30         {
31             if (sm > a[i][j])
32                 p = j;
33         }
34         large = a[i][p];
35         for (j = 0; j < num; j++)
36         {
37             if (large < a[j][p])
38                 f = 0;
39         }
40         if (f == 0)
41             break;
42     }
43     if (f == 1)
44         printf("The Saddle Point is at (%d, %d)", i, p);
45     else
46         printf("There is no saddle point in the matrix.");
47     return 0;
48 }
```

Enter Size of Matrix : 3
Enter Matrix Elements :
3 6 9
2 5 8
1 4 7
The Entered Matrix is :
3 6 9
2 5 8
1 4 7
The Saddle Point for the given Matrix is : 3

Process exited after 11.52 seconds with return value 0
Press any key to continue . . .

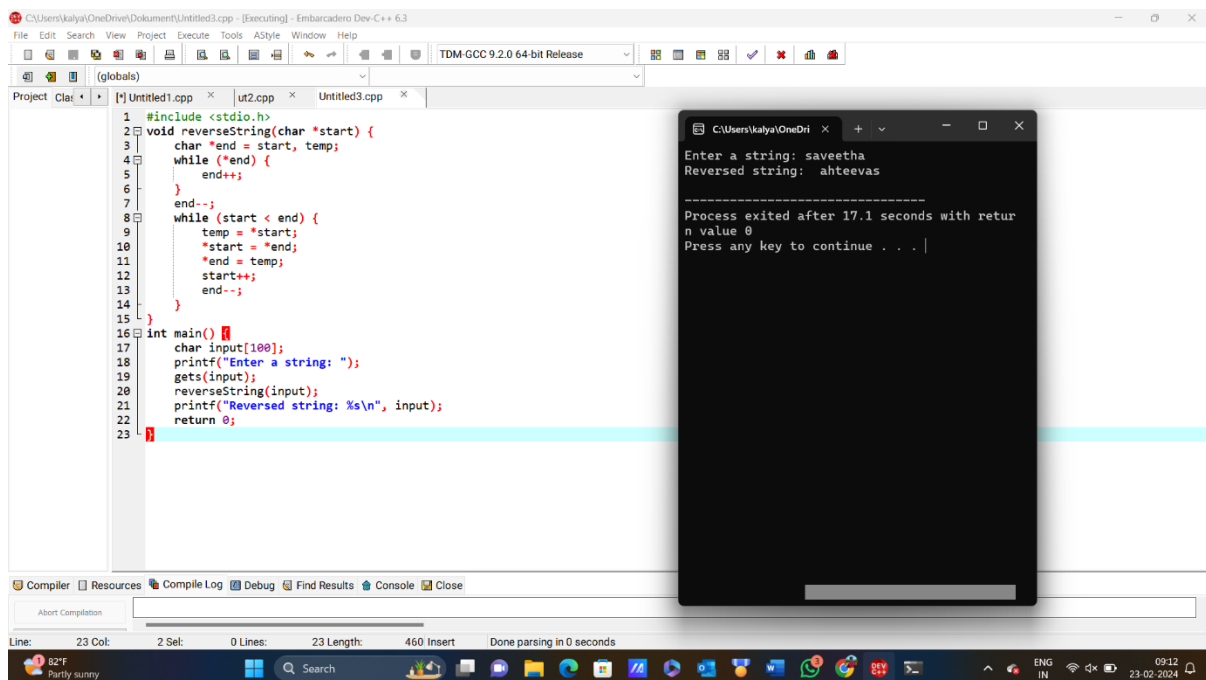
2. find the length of string using pointers .



```
1 #include <stdio.h>
2 int getLength(char *p)
3 {
4     int count = 0;
5     while (*p != '\0')
6     {
7         count++;
8         p++;
9     }
10    return count;
11 }
12 int main()
13 {
14     char str[20] = "savetha";
15     int length = 0;
16     length = getLength(str);
17     printf("The length of the string \"%s\" is %d", str, length);
18     return 0;
19 }
```

Output
/tmp/3f9zz29DLL.o
The length of the string "savetha" is 7

3. reversing the given string using pointers



The screenshot shows an IDE with a C++ program that reverses a string using pointers. The code is as follows:

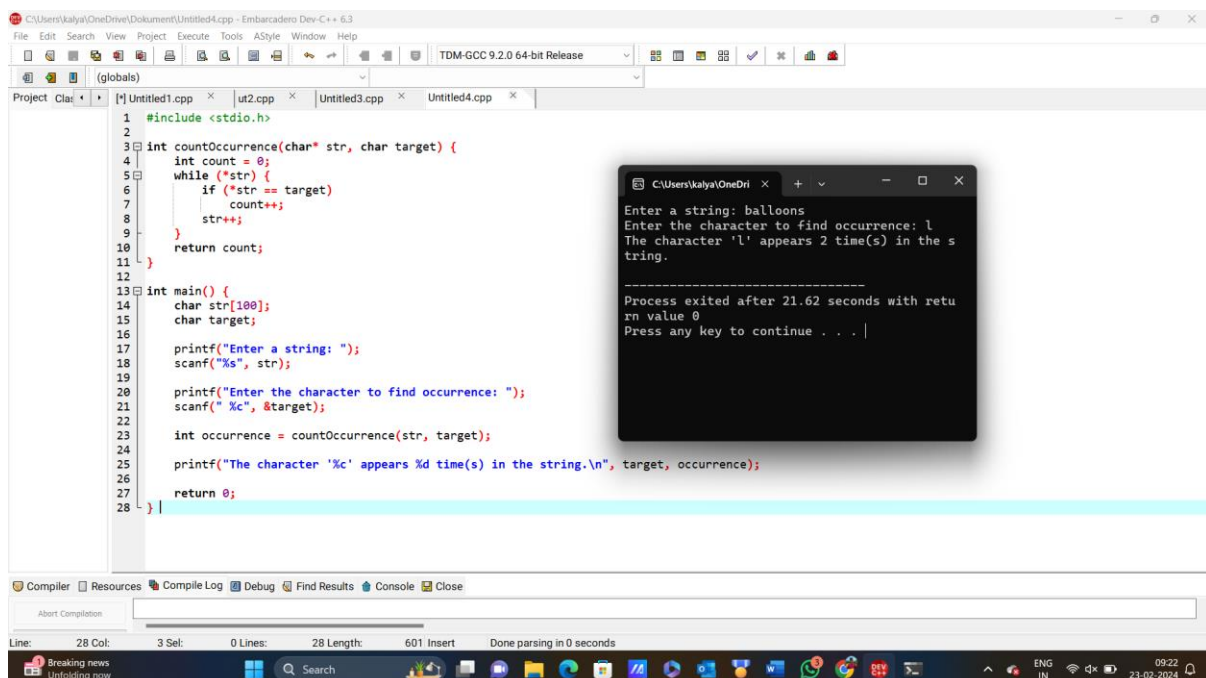
```
1 #include <stdio.h>
2 void reverseString(char *start) {
3     char *end = start; temp;
4     while (*end) {
5         end++;
6     }
7     end--;
8     while (start < end) {
9         temp = *start;
10        *start = *end;
11        *end = temp;
12        start++;
13        end--;
14    }
15 }
16 int main() {
17     char input[100];
18     printf("Enter a string: ");
19     gets(input);
20     reverseString(input);
21     printf("Reversed string: %s\n", input);
22     return 0;
23 }
```

The execution output is shown in a terminal window:

```
Enter a string: saveetha
Reversed string: ahteervas

-----
Process exited after 17.1 seconds with return value 0
Press any key to continue . . .
```

4.find the number of occurrence of elements in a string.



The screenshot shows an IDE with a C++ program that counts the occurrence of a character in a string. The code is as follows:

```
1 #include <stdio.h>
2
3 int countOccurrence(char* str, char target) {
4     int count = 0;
5     while (*str) {
6         if (*str == target)
7             count++;
8         str++;
9     }
10    return count;
11 }
12
13 int main() {
14     char str[100];
15     char target;
16
17     printf("Enter a string: ");
18     scanf("%s", str);
19
20     printf("Enter the character to find occurrence: ");
21     scanf("%c", &target);
22
23     int occurrence = countOccurrence(str, target);
24
25     printf("The character '%c' appears %d time(s) in the string.\n", target, occurrence);
26
27     return 0;
28 }
```

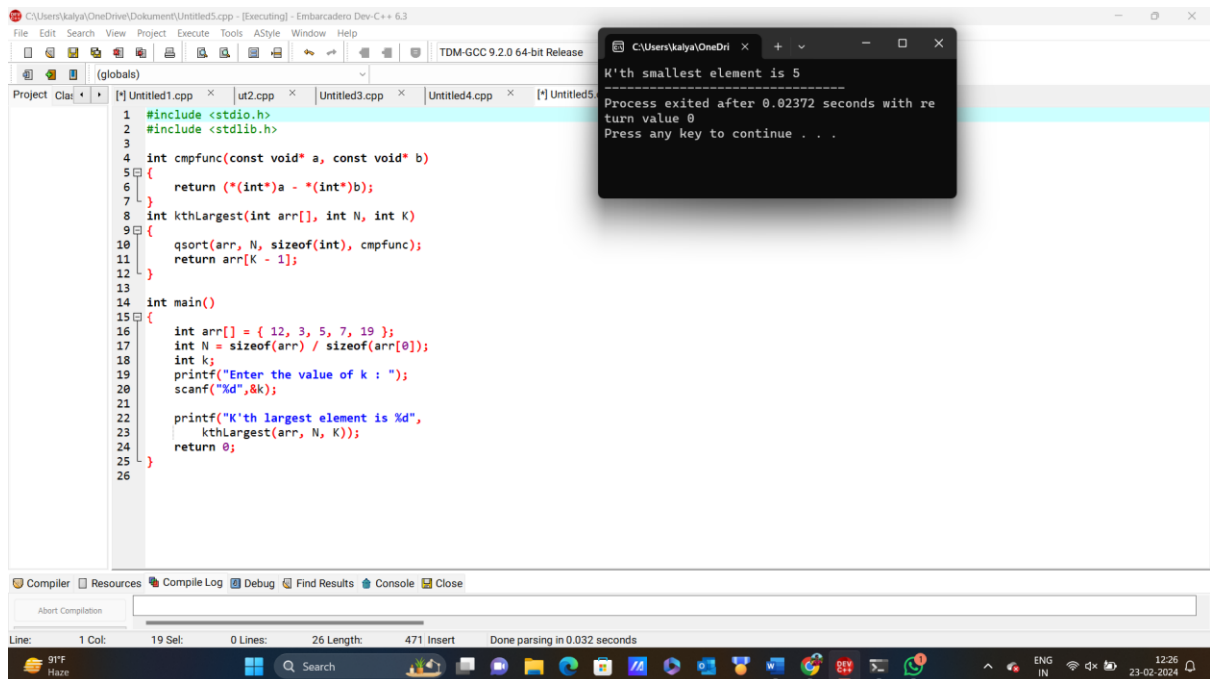
The execution output is shown in a terminal window:

```
Enter a string: balloons
Enter the character to find occurrence: l
The character 'l' appears 2 time(s) in the string.

-----
Process exited after 21.62 seconds with return value 0
Press any key to continue . . .
```

Analytical session Day – 3:

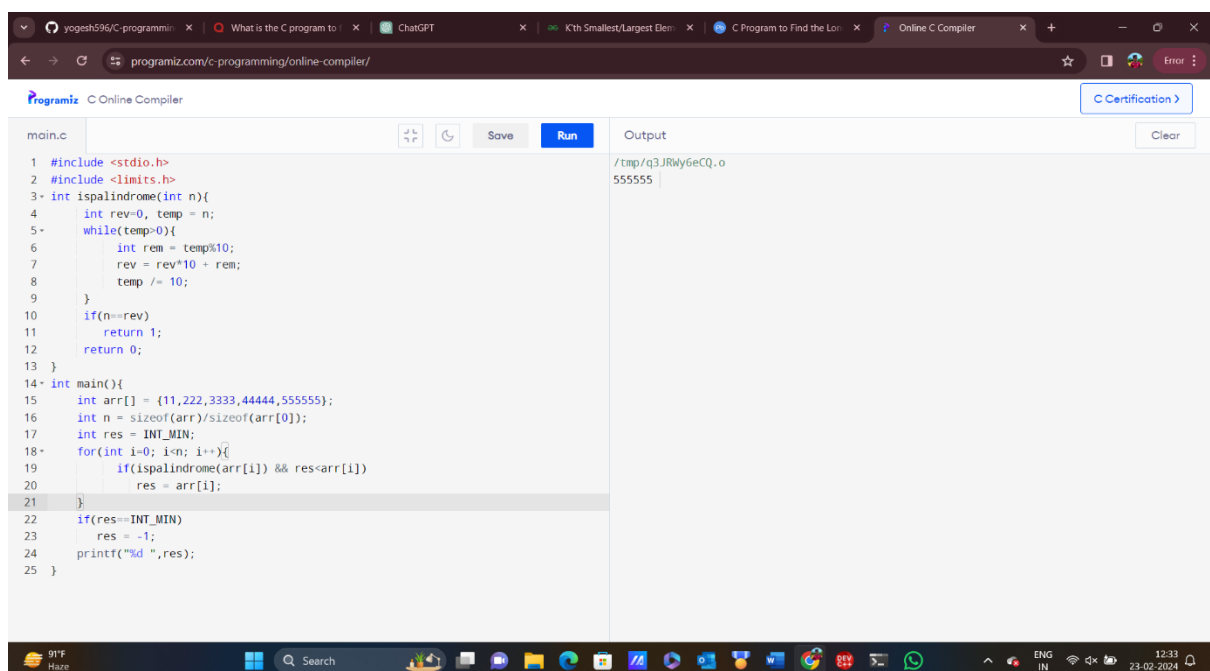
1.finding a largest kth element in an array.



```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int cmpfunc(const void* a, const void* b)
5 {
6     return (*(int*)a - *(int*)b);
7 }
8 int kthLargest(int arr[], int N, int K)
9 {
10     qsort(arr, N, sizeof(int), cmpfunc);
11     return arr[K - 1];
12 }
13
14 int main()
15 {
16     int arr[] = { 12, 3, 5, 7, 19 };
17     int N = sizeof(arr) / sizeof(arr[0]);
18     int k;
19     printf("Enter the value of k : ");
20     scanf("%d", &k);
21
22     printf("K'th largest element is %d",
23           kthLargest(arr, N, K));
24     return 0;
25 }
```

K'th smallest element is 5
Process exited after 0.02372 seconds with return value 0
Press any key to continue . . .

2.Find the Longest Palindrome in an Array.



```
1 #include <stdio.h>
2 #include <limits.h>
3 int ispalindrome(int n){
4     int rev=0, temp = n;
5     while(temp>0){
6         int rem = temp%10;
7         rev = rev*10 + rem;
8         temp /= 10;
9     }
10    if(n==rev)
11        return 1;
12    return 0;
13 }
14 int main(){
15     int arr[] = {11,222,3333,4444,555555};
16     int n = sizeof(arr)/sizeof(arr[0]);
17     int res = INT_MIN;
18     for(int i=0; i<n; i++){
19         if(ispalindrome(arr[i]) && res<arr[i])
20             res = arr[i];
21     }
22     if(res==INT_MIN)
23         res = -1;
24     printf("%d ",res);
25 }
```

/tmp/q3JRwy6eCQ.o
555555

