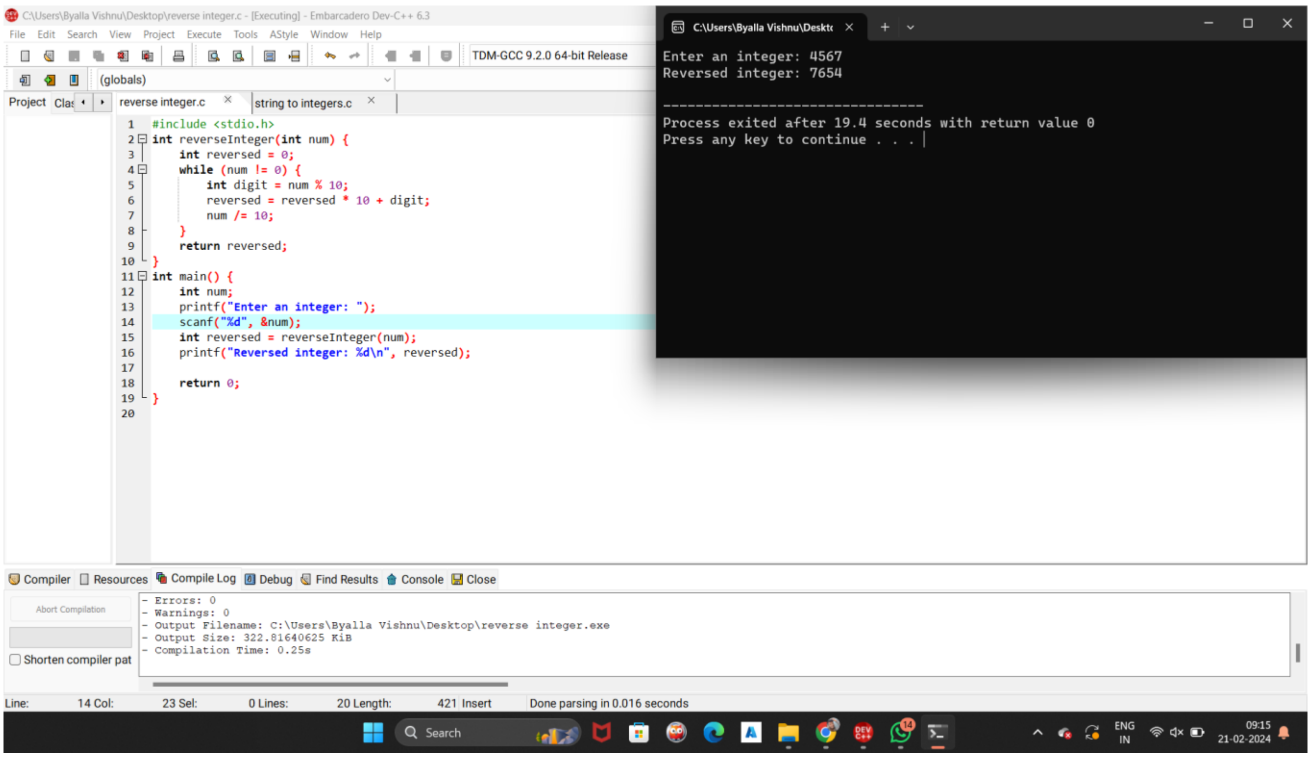
C programing

Day-1

K Yogesh.

192210596.

1 program -reverse integer



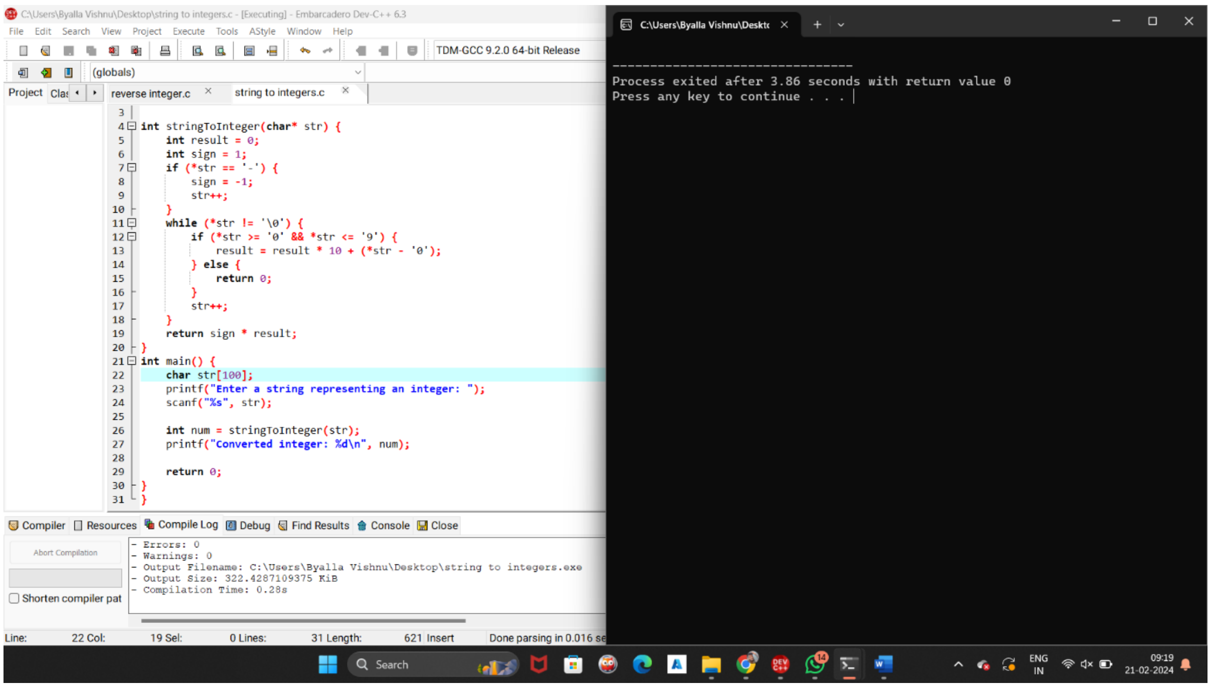
```c
#include <stdio.h>
int reverseInteger(int num) {
    int reversed = 0;
    while (num != 0) {
        int digit = num % 10;
        reversed = reversed * 10 + digit;
        num /= 10;
    }
    return reversed;
}
int main() {
    int num;
    printf("Enter an integer: ");
    scanf("%d", &num);
    int reversed = reverseInteger(num);
    printf("Reversed integer: %d\n", reversed);

    return 0;
}
```

```
Enter an integer: 4567
Reversed integer: 7654

--------------------------------
Process exited after 19.4 seconds with return value 0
Press any key to continue . . .
```

```
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\Byalla Vishnu\Desktop\reverse integer.exe
- Output Size: 322.81640625 KiB
- Compilation Time: 0.25s
```

2 program

String to integers


C:\Users\Byalla Vishnu\Desktop\string to integers.c - [Executing] - Embarcadero Dev-C++ 6.3

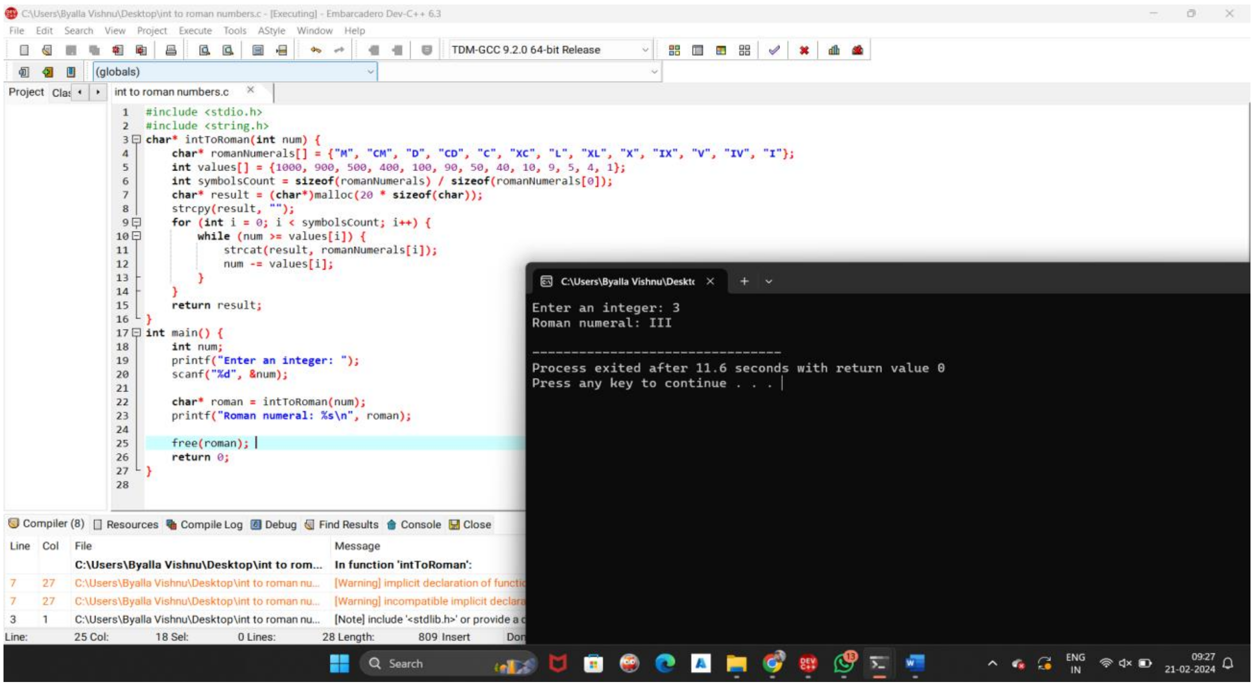
```c

int stringToInteger(char* str) {
    int result = 0;
    int sign = 1;
    if (*str == '-') {
        sign = -1;
        str++;
    }
    while (*str != '\0') {
        if (*str >= '0' && *str <= '9') {
            result = result * 10 + (*str - '0');
        } else {
            return 0;
        }
        str++;
    }
    return sign * result;
}
int main() {
    char str[100];
    printf("Enter a string representing an integer: ");
    scanf("%s", str);

    int num = stringToInteger(str);
    printf("Converted integer: %d\n", num);

    return 0;
}
}
```

```
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\Byalla Vishnu\Desktop\string to integers.exe
- Output Size: 322.4287109375 KiB
- Compilation Time: 0.28s
```

```
--------------------------------
Process exited after 3.86 seconds with return value 0
Press any key to continue . . .
```

Program 3

Integer to roman


C:\Users\Byalla Vishnu\Desktop\int to roman numbers.c - [Executing] - Embarcadero Dev-C++ 6.3

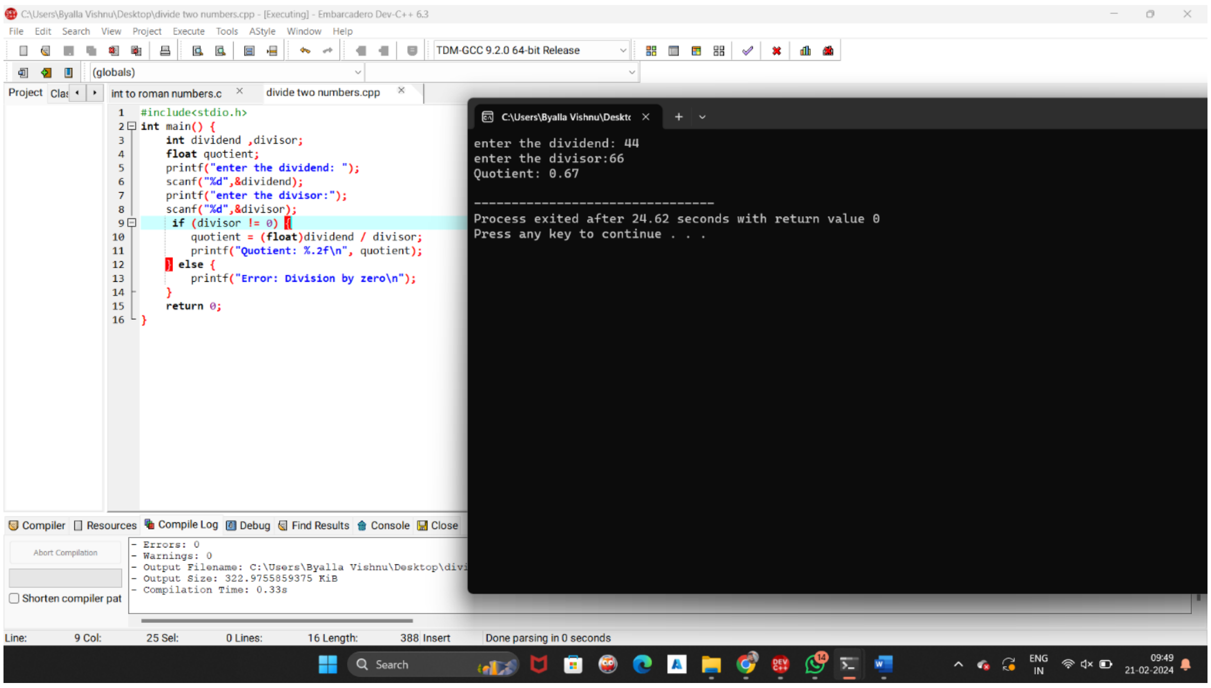
```c
#include <stdio.h>
#include <string.h>
char* intToRoman(int num) {
    char* romanNumerals[] = {"M", "CM", "D", "CD", "C", "XC", "L", "XL", "X", "IX", "V", "IV", "I"};
    int values[] = {1000, 900, 500, 400, 100, 90, 50, 40, 10, 9, 5, 4, 1};
    int symbolsCount = sizeof(romanNumerals) / sizeof(romanNumerals[0]);
    char* result = (char*)malloc(20 * sizeof(char));
    strcpy(result, "");
    for (int i = 0; i < symbolsCount; i++) {
        while (num >= values[i]) {
            strcat(result, romanNumerals[i]);
            num -= values[i];
        }
    }
    return result;
}
int main() {
    int num;
    printf("Enter an integer: ");
    scanf("%d", &num);

    char* roman = intToRoman(num);
    printf("Roman numeral: %s\n", roman);

    free(roman);
    return 0;
}
```

```
Line  Col  File                                           Message
           C:\Users\Byalla Vishnu\Desktop\int to rom...   In function 'intToRoman':
7     27   C:\Users\Byalla Vishnu\Desktop\int to roman nu...  [Warning] implicit declaration of functi
7     27   C:\Users\Byalla Vishnu\Desktop\int to roman nu...  [Warning] incompatible implicit declara
3     1    C:\Users\Byalla Vishnu\Desktop\int to roman nu...  [Note] include '<stdlib.h>' or provide a c
```

```
Enter an integer: 3
Roman numeral: III

--------------------------------
Process exited after 11.6 seconds with return value 0
Press any key to continue . . .
```

4- problem

Divide two numbers



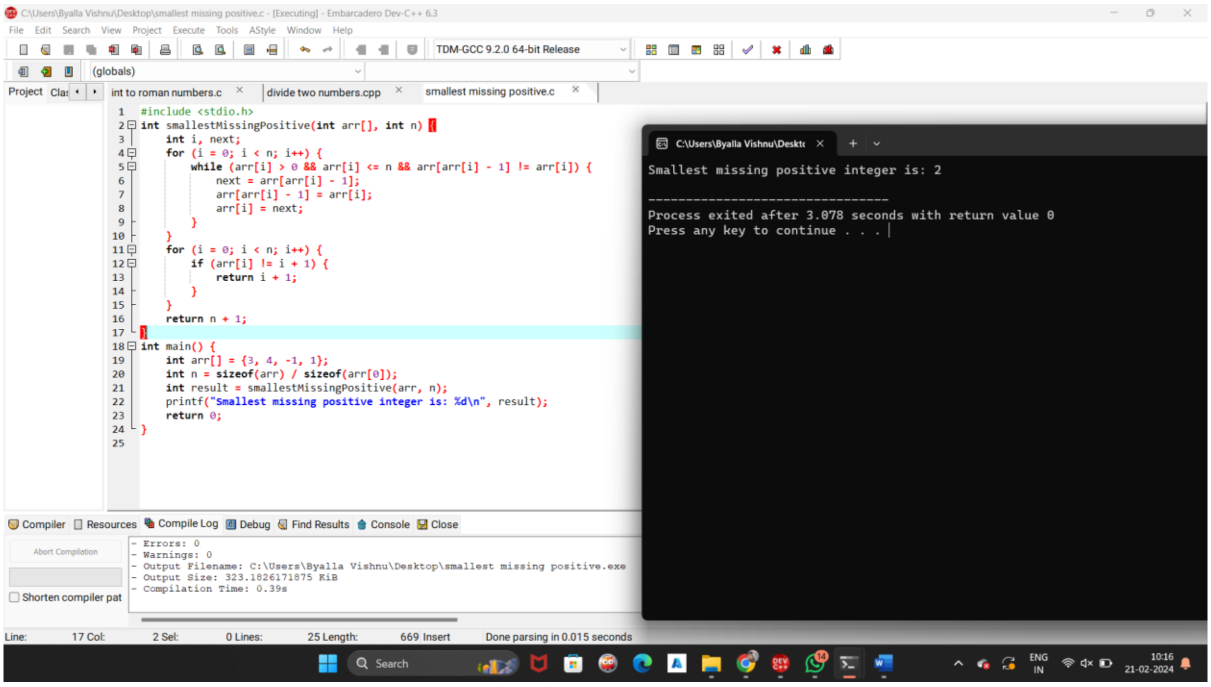
```
#include<stdio.h>
int main() {
    int dividend ,divisor;
    float quotient;
    printf("enter the dividend: ");
    scanf("%d",&dividend);
    printf("enter the divisor:");
    scanf("%d",&divisor);
     if (divisor != 0) {
        quotient = (float)dividend / divisor;
        printf("Quotient: %.2f\n", quotient);
    } else {
        printf("Error: Division by zero\n");
    }
    return 0;
}
```

```
enter the dividend: 44
enter the divisor:66
Quotient: 0.67

--------------------------------
Process exited after 24.62 seconds with return value 0
Press any key to continue . . .
```

```
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\Byalla Vishnu\Desktop\divi
- Output Size: 322.9755859375 KiB
- Compilation Time: 0.33s
```

5- problem

Smallest missing positive integer in an array of integers



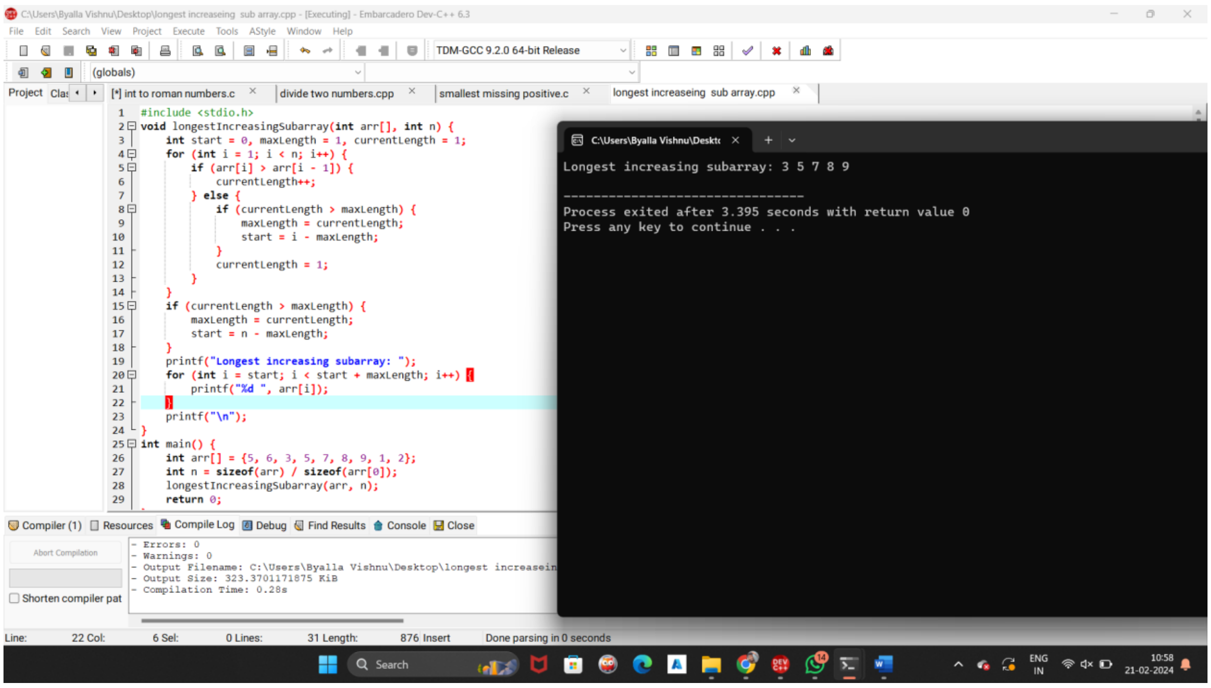
```
#include <stdio.h>
int smallestMissingPositive(int arr[], int n) {
    int i, next;
    for (i = 0; i < n; i++) {
        while (arr[i] > 0 && arr[i] <= n && arr[arr[i] - 1] != arr[i]) {
            next = arr[arr[i] - 1];
            arr[arr[i] - 1] = arr[i];
            arr[i] = next;
        }
    }
    for (i = 0; i < n; i++) {
        if (arr[i] != i + 1) {
            return i + 1;
        }
    }
    return n + 1;
}
int main() {
    int arr[] = {3, 4, -1, 1};
    int n = sizeof(arr) / sizeof(arr[0]);
    int result = smallestMissingPositive(arr, n);
    printf("Smallest missing positive integer is: %d\n", result);
    return 0;
}
```

```
Smallest missing positive integer is: 2

--------------------------------
Process exited after 3.078 seconds with return value 0
Press any key to continue . . .
```

```
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\Byalla Vishnu\Desktop\smallest missing positive.exe
- Output Size: 323.1826171875 KiB
- Compilation Time: 0.39s
```

6 program

longest increasing subarray.



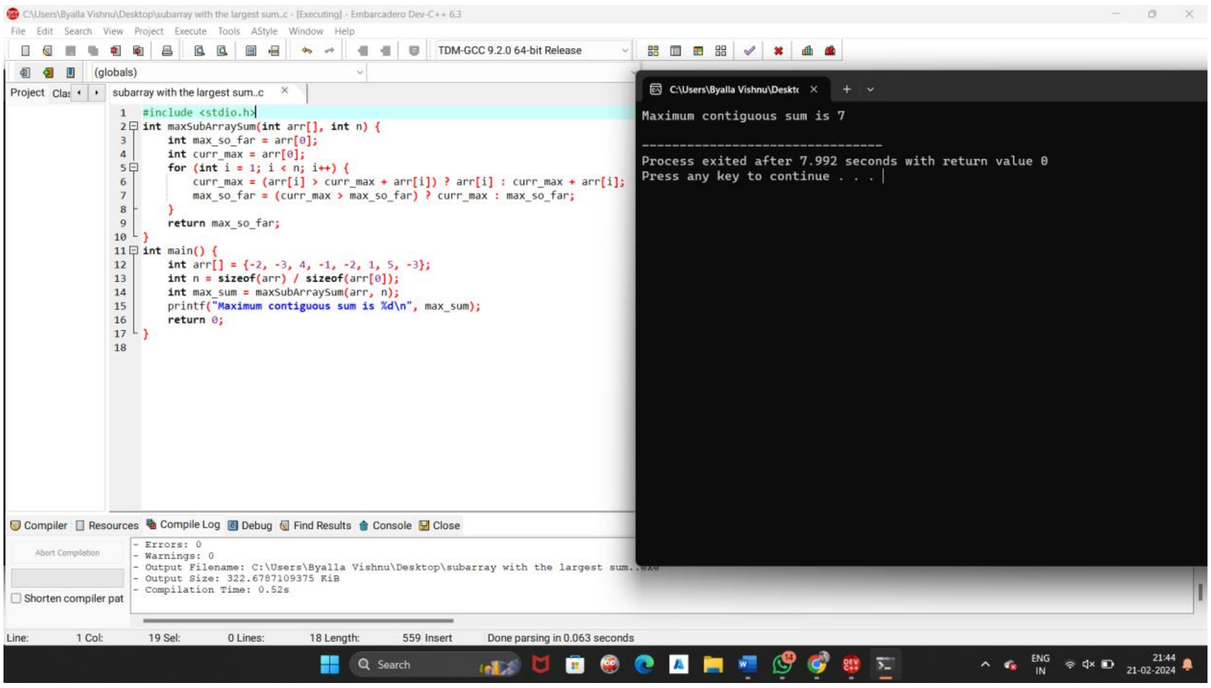
```c
#include <stdio.h>
void longestIncreasingSubarray(int arr[], int n) {
    int start = 0, maxLength = 1, currentLength = 1;
    for (int i = 1; i < n; i++) {
        if (arr[i] > arr[i - 1]) {
            currentLength++;
        } else {
            if (currentLength > maxLength) {
                maxLength = currentLength;
                start = i - maxLength;
            }
            currentLength = 1;
        }
    }
    if (currentLength > maxLength) {
        maxLength = currentLength;
        start = n - maxLength;
    }
    printf("Longest increasing subarray: ");
    for (int i = start; i < start + maxLength; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
}
int main() {
    int arr[] = {5, 6, 3, 5, 7, 8, 9, 1, 2};
    int n = sizeof(arr) / sizeof(arr[0]);
    longestIncreasingSubarray(arr, n);
    return 0;
```

```
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\Byalla Vishnu\Desktop\longest increasein
- Output Size: 323.3701171875 KiB
- Compilation Time: 0.28s
```

```
Longest increasing subarray: 3 5 7 8 9

--------------------------------
Process exited after 3.395 seconds with return value 0
Press any key to continue . . .
```

7 program

subarray with the largest sum.



```c
#include <stdio.h>
int maxSubArraySum(int arr[], int n) {
    int max_so_far = arr[0];
    int curr_max = arr[0];
    for (int i = 1; i < n; i++) {
        curr_max = (arr[i] > curr_max + arr[i]) ? arr[i] : curr_max + arr[i];
        max_so_far = (curr_max > max_so_far) ? curr_max : max_so_far;
    }
    return max_so_far;
}
int main() {
    int arr[] = {-2, -3, 4, -1, -2, 1, 5, -3};
    int n = sizeof(arr) / sizeof(arr[0]);
    int max_sum = maxSubArraySum(arr, n);
    printf("Maximum contiguous sum is %d\n", max_sum);
    return 0;
}
```

```
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\Byalla Vishnu\Desktop\subarray with the largest sum..exe
- Output Size: 322.6787109375 KiB
- Compilation Time: 0.52s
```

```
Maximum contiguous sum is 7

--------------------------------
Process exited after 7.992 seconds with return value 0
Press any key to continue . . .
```