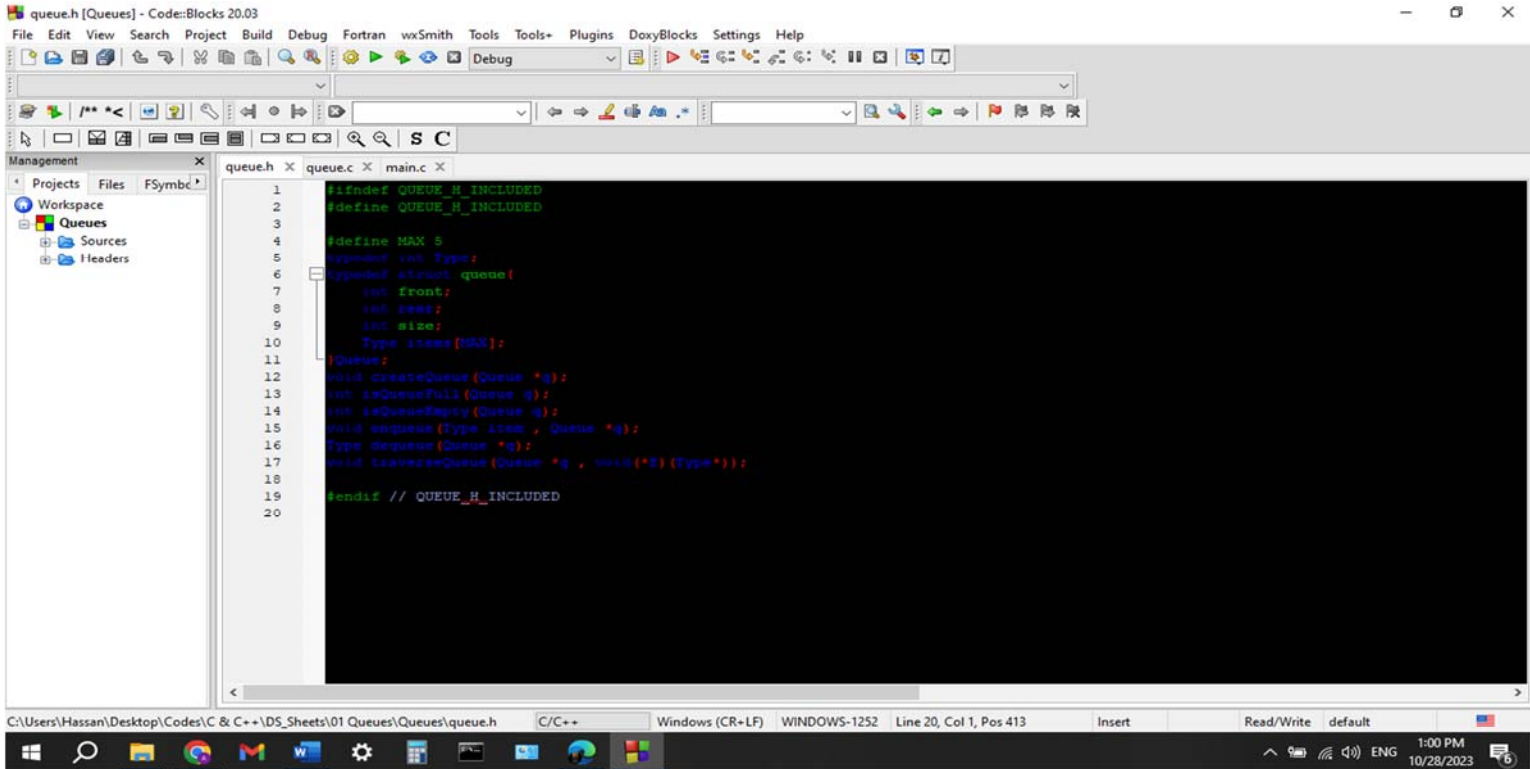


Sheet #2 – Ahmed Hassan Ahmed Emara – 20220017

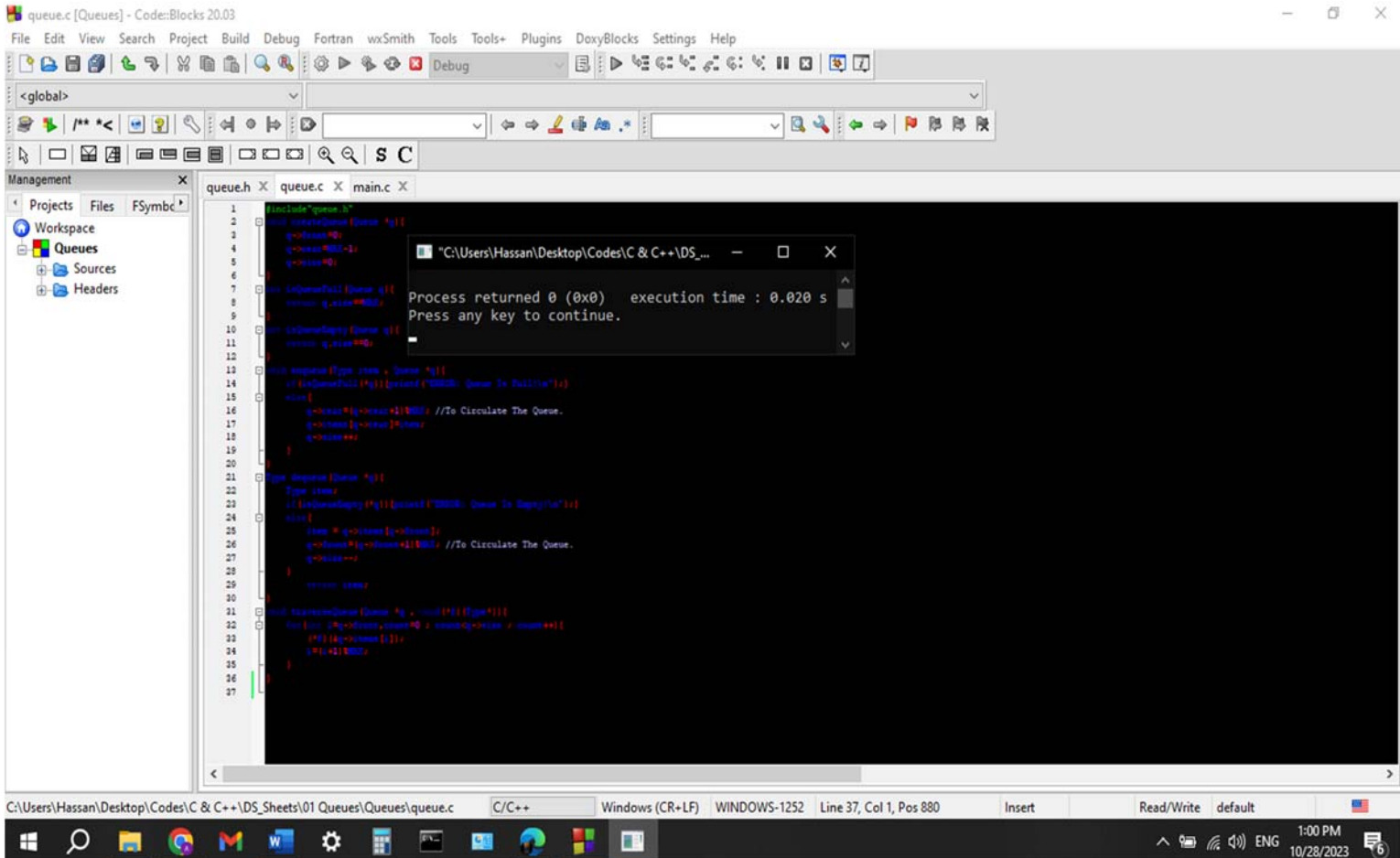
1) 1-



The screenshot shows the Code::Blocks IDE with the 'queue.h' header file open. The file contains the following code:

```
1 #ifndef QUEUE_H_INCLUDED
2 #define QUEUE_H_INCLUDED
3
4 #define MAX 5
5
6 typedef struct queue{
7     int front;
8     int rear;
9     int size;
10    Type items[MAX];
11 }Queue;
12
13 void createQueue(Queue *q);
14 int isQueueFull(Queue q);
15 int isQueueEmpty(Queue q);
16 void enqueue(Type item , Queue *q);
17 Type dequeue(Queue *q);
18 void traverseQueue(Queue *q , void (*f)(Type));
19
20 #endif // QUEUE_H_INCLUDED
```

The status bar at the bottom indicates the file path: C:\Users\Hassan\Desktop\Codes\C & C++\DS_Sheets\01 Queues\Queues\queue.h, and the cursor is at Line 20, Col 1, Pos 413.



The screenshot shows the Code::Blocks IDE with the 'queue.c' source file open. The file contains the following code:

```
1 #include "queue.h"
2 void createQueue(Queue *q){
3     q->front=0;
4     q->rear=0;
5     q->size=0;
6 }
7 int isQueueFull(Queue q){
8     return q->size==MAX;
9 }
10 int isQueueEmpty(Queue q){
11     return q->size==0;
12 }
13 void enqueue(Type item , Queue *q){
14     if(isQueueFull(*q))printf("ERROR: Queue Is Full!\n");
15     else{
16         q->items[q->rear]=item; //To Circulate The Queue.
17         q->size++;q->rear++;
18     }
19 }
20 Type dequeue(Queue *q){
21     Type item;
22     if(isQueueEmpty(*q))printf("ERROR: Queue Is Empty!\n");
23     else{
24         item = q->items[q->front];
25         q->items[q->front]=q->items[q->rear]; //To Circulate The Queue.
26         q->size--;q->front++;
27     }
28     return item;
29 }
30
31 void traverseQueue(Queue *q , void (*f)(Type)){
32     for(int i=q->front; i<=q->rear; i++){
33         f(q->items[i]);
34     }
35 }
```

The execution output is displayed in a small window titled "C:\Users\Hassan\Desktop\Codes\C & C++\DS_...". It shows: "Process returned 0 (0x0) execution time : 0.020 s Press any key to continue."

The status bar at the bottom indicates the file path: C:\Users\Hassan\Desktop\Codes\C & C++\DS_Sheets\01 Queues\Queues\queue.c, and the cursor is at Line 37, Col 1, Pos 880.

2-

```
#include <stdio.h>
#include "queue1.h"

int main() {
    Queue q;
    Type item;
    createQueue(&q);
    char choice;
    enqueue(1,&q);
    enqueue(2,&q);
    enqueue(3,&q);
    enqueue(4,&q);
    printf("(a) Read an element then enqueue it.\n");
    printf("(b) Dequeue an element then display it.\n");
    printf("(c) Exit.\n");
    scanf("%c",&choice);
    switch(choice) {
        case 'a': printf("Enter the element: ");
                 scanf("%d",&item);
                 enqueue(item,&q);
                 break;
        case 'b': printf("%d",dequeue(&q));break;
        default: return 0;
    }
}
```

Process returned 0 (0x0) execution time : 1.637 s
Press any key to continue.

Yes, because it is executable.

3-

```
int queue_size(Queue q) {
    return q.rear+1;
}
```

```
#include <stdio.h>
#include "queue1.h"

int main() {
    Queue q;
    Type item;
    createQueue(&q);
    char choice;
    enqueue(1,&q);
    enqueue(2,&q);
    enqueue(3,&q);
    enqueue(4,&q);
    enqueue(5,&q);
    printf("(a) Read an element then enqueue it.\n");
    printf("(b) Dequeue an element then display it.\n");
    printf("(c) Show pre- and post-conditions of queue_size function.\n");
    printf("(d) Use queue_size function.\n");
    printf("(e) Exit.\n");
    scanf("%c",&choice);
    switch(choice) {
        case 'a': printf("Enter the element: ");
                 scanf("%d",&item);
                 enqueue(item,&q);
                 break;
        case 'b': printf("%d",dequeue(&q));break;
        case 'c': printf("Pre- and post-conditions of queue_size function.\n");break;
        case 'd': printf("%d",queue_size(q));break;
        default: return 0;
    }
}
```

Process returned 0 (0x0) execution time : 3.224 s
Press any key to continue.

4-

```

1 #include <stdio.h>
2 #include "queue1.h"
3 void content(Queue q) {
4     Type item;
5     while (!isQueueEmpty(q)) {
6         printf("%d\n", dequeue(&q));
7     }
8 }
9
10 int main() {
11     Queue q;
12     Type item;
13     createQueue(&q);
14     enqueue(1, &q);
15     enqueue(2, &q);
16     enqueue(3, &q);
17     enqueue(4, &q);
18     enqueue(5, &q);
19     content(q);
20 }

```

Because we are supposed to know nothing about queue.c .

2)

```

int queue_size(Queue q) {
    return q.rear+1;
}

```

```

1 #include <stdio.h>
2 #include "queue1.h"
3 int main() {
4     Queue q;
5     Type item;
6     createQueue(&q);
7     enqueue(10, &q);
8     enqueue(20, &q);
9     enqueue(30, &q);
10    enqueue(40, &q);
11    enqueue(50, &q);
12    printf("%d\n", last(&q));
13 }
14

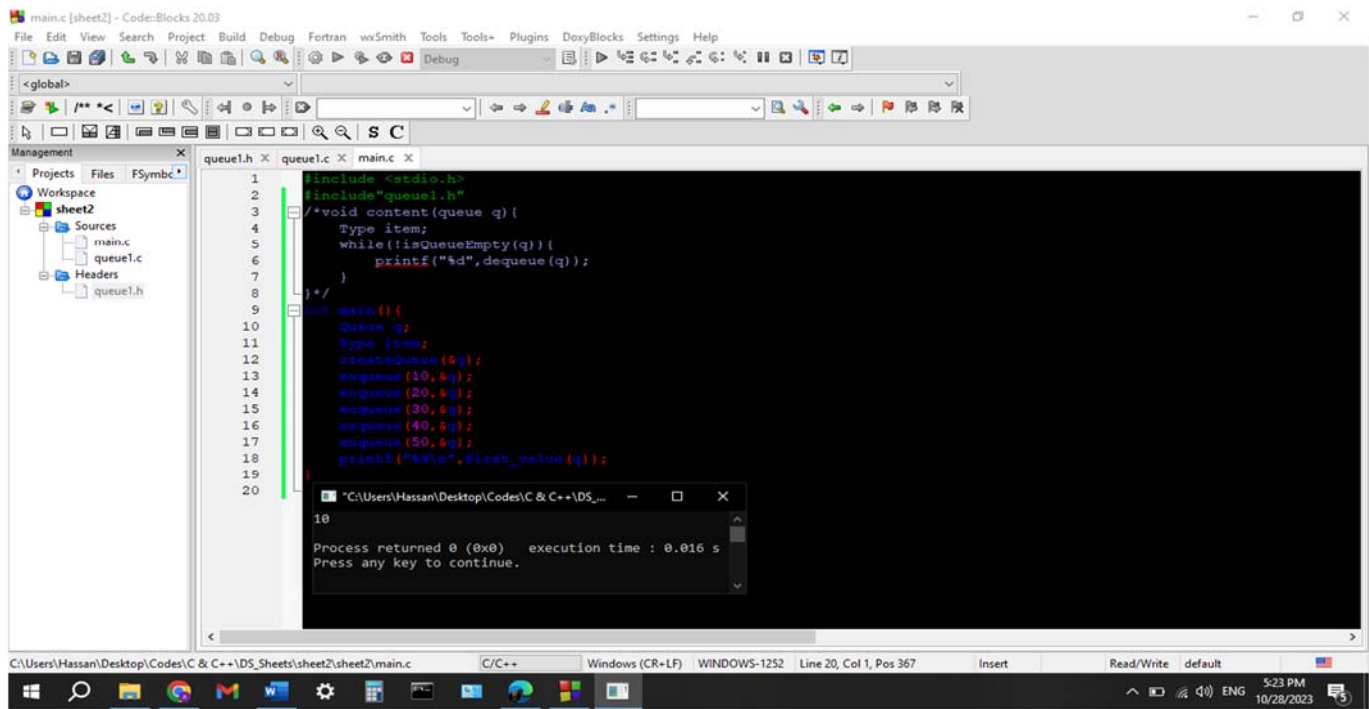
```

3)

```

Type first_value(Queue q){
    if(isQueueEmpty(q)){printf("ERROR: Queue Is Empty!");}
    else{return q.items[q.front];}
}

```

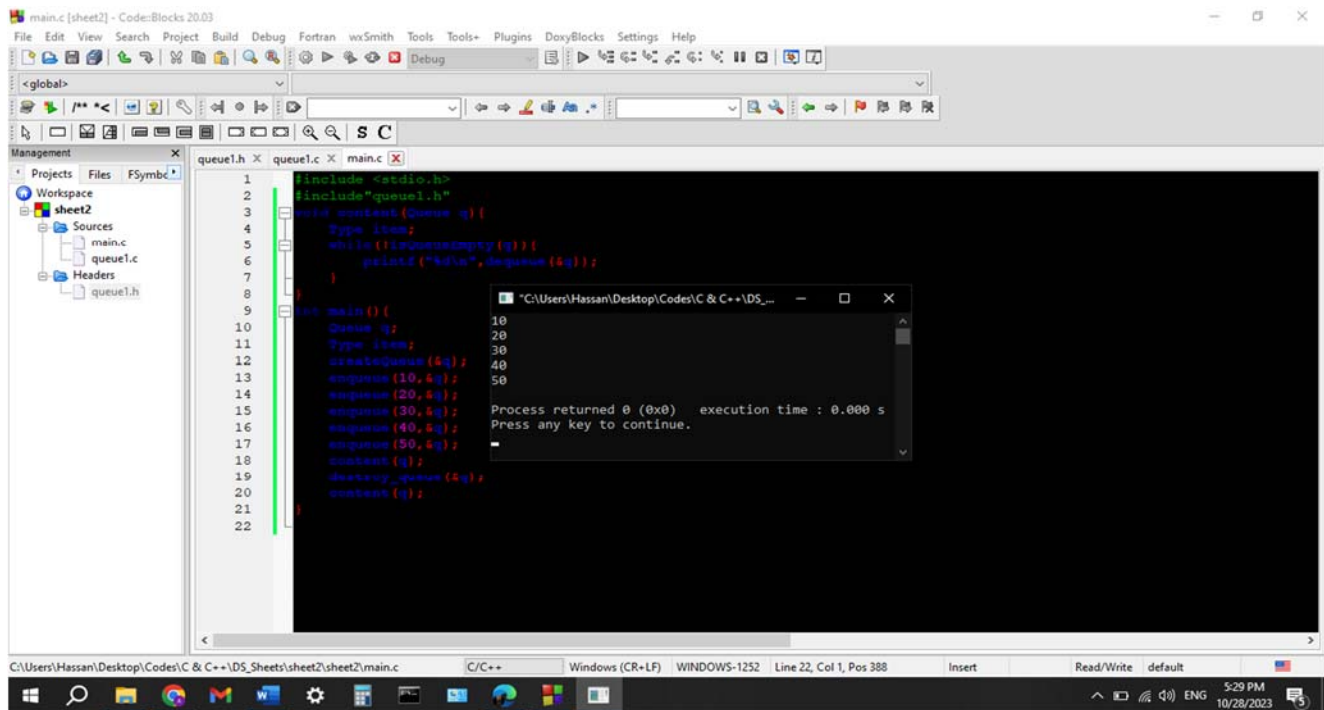


4)

```

void destroy_queue(Queue *q){
    while(!isQueueEmpty(*q)){
        q->front=(q->front+1)%MAX;
        q->size--;
    }
}

```



5)

The screenshot shows the Code::Blocks IDE with the file `queue1.c` open. The code implements a queue structure and a function `cpy_queue`. The queue is represented as an array with a counter to track the current position. The `cpy_queue` function copies the contents of one queue into another.

```
64     }
65 }
66 void cpy_queue(Queue q) {
67     Queue k;
68     Type item;
69     createQueue(&k);
70     Type arr[MAX];
71     int counter=0;
72     while(!isQueueEmpty(q)){
73         arr[counter]=dequeue(&q);counter++;
74     }
75     counter=0;
76     while(!isQueueFull(k)){
77         enqueue(arr[counter] , &k);counter++;
78     }
79     counter=0;
80     while(!(counter==MAX)){
81         printf("%d\n" , k.items[counter]);counter++;
82     }
83 }
84 }
```

The status bar at the bottom indicates the file path: `C:\Users\Hassan\Desktop\Codes\C & C++\DS_Sheets\sheet2\sheet2\queue1.c`, the compiler is `C/C++`, and the window title is `Windows (CR+LF) WINDOWS-1252`. The cursor is at line 84, column 1, position 1955.

The screenshot shows the Code::Blocks IDE with the file `main.c` open. The code includes `queue1.h` and implements the `main` function. It creates a queue, enqueues several values, and then calls `cpy_queue` to copy the queue. The execution output is shown in a separate window.

```
1 #include <stdio.h>
2 #include "queue1.h"
3 void content(Queue q) {
4     Type item;
5     while(!isQueueEmpty(q)){
6         printf("%d\n",dequeue(&q));
7     }
8 }
9 int main() {
10     Queue q;
11     Type item;
12     createQueue(&q);
13     enqueue(10,&q);
14     enqueue(20,&q);
15     enqueue(30,&q);
16     enqueue(40,&q);
17     enqueue(50,&q);
18     cpy_queue(q);
19 }
20 }
```

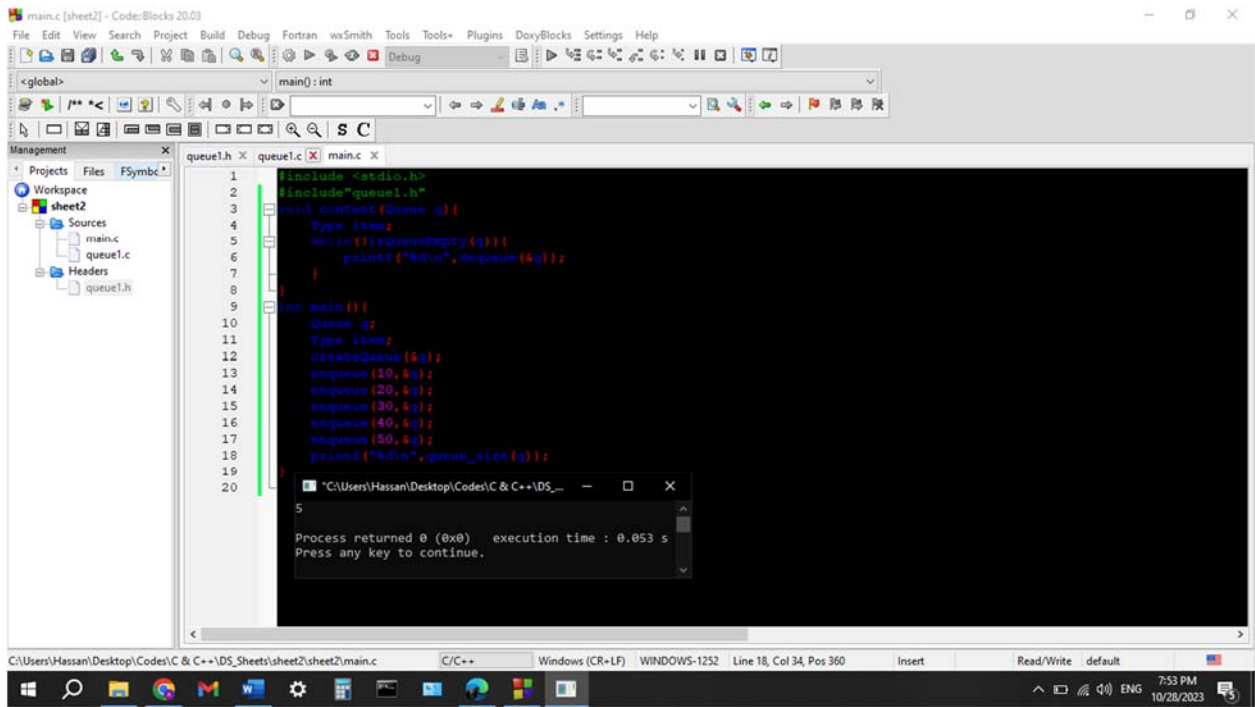
The execution output window shows the following text:

```
10
20
30
40
50
Process returned 0 (0x0)   execution time : 0.069 s
Press any key to continue.
```

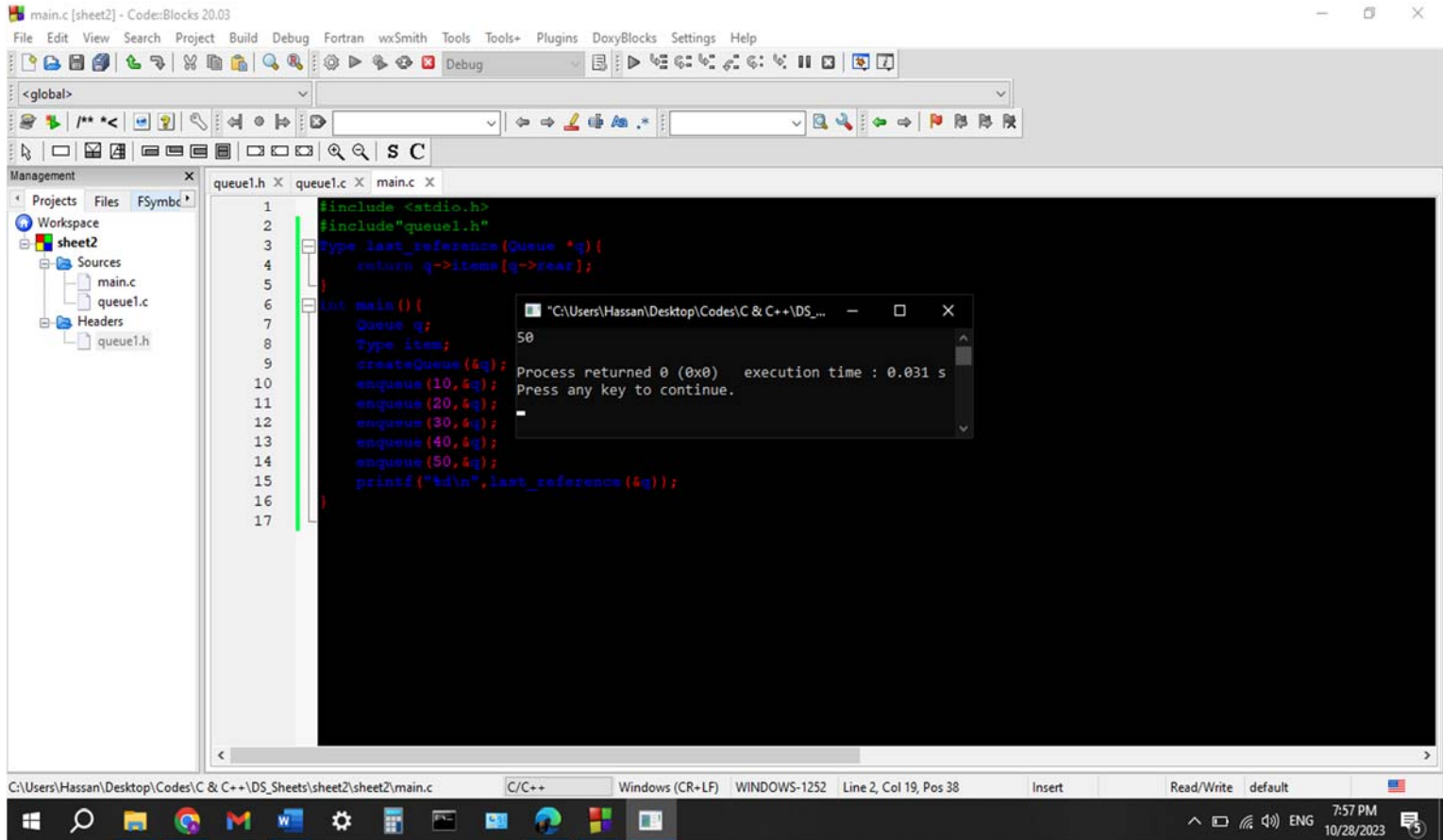
The status bar at the bottom indicates the file path: `C:\Users\Hassan\Desktop\Codes\C & C++\DS_Sheets\sheet2\sheet2\main.c`, the compiler is `C/C++`, and the window title is `Windows (CR+LF) WINDOWS-1252`. The cursor is at line 18, column 18, position 344.

6)

```
int queue_size(Queue q){
    return q.rear+1;
}
```



7)



8)

```

1 #include <stdio.h>
2 #include "queue1.h"
3
4 Type first(Queue q) {
5     if (isQueueEmpty(q)) { printf("ERROR: Queue is Empty!"); }
6     else { return q.items[q.front]; }
7 }
8
9 int main() {
10     Queue q;
11     Type item;
12     createQueue(&q);
13     enqueue(10, &q);
14     enqueue(20, &q);
15     enqueue(30, &q);
16     enqueue(40, &q);
17     enqueue(50, &q);
18     printf("%d\n", first(q));
19 }

```

Process returned 0 (0x0) execution time : 0.108 s
Press any key to continue.

9)

```

1 #include <stdio.h>
2 #include "queue1.h"
3
4 void destroy(Queue *q) {
5     while (!isQueueEmpty(*q)) {
6         q->front = (q->front + 1) % MAX;
7         q->size--;
8     }
9 }
10
11 int main() {
12     Queue q;
13     Type item;
14     createQueue(&q);
15     enqueue(10, &q);
16     enqueue(20, &q);
17     enqueue(30, &q);
18     enqueue(40, &q);
19     enqueue(50, &q);
20     printf("%d\n", first(q));
21     dequeue(&q);
22     printf("%d\n", first(q));
23     destroy(&q);
24 }

```

Process returned 0 (0x0) execution time : 0.073 s
Press any key to continue.

10)

```

1 #include <stdio.h>
2 #include "queue1.h"
3 int main() {
4     Queue q;
5     Type item;
6     createQueue(&q);
7     enqueue(10, &q);
8     enqueue(20, &q);
9     enqueue(30, &q);
10    enqueue(40, &q);
11    enqueue(50, &q);
12    printf("%d", size(q));
13    return 0;
14 }

```

Process returned 0 (0x0) execution time : 0.032 s
Press any key to continue.

11)

```

1 #include <stdio.h>
2 #include "queue1.h"
3 int size(Queue q) {
4     return q.rear+1;
5 }
6 int main() {
7     Queue q;
8     Type item;
9     createQueue(&q);
10    enqueue(10, &q);
11    enqueue(20, &q);
12    enqueue(30, &q);
13    enqueue(40, &q);
14    enqueue(50, &q);
15    printf("%d", size(q));
16 }

```

Process returned 0 (0x0) execution time : 0.037 s
Press any key to continue.

12)

```

stack1.h X stack1.c X test.c X
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <stdbool.h>
4  #include <string.h>
5
6  #define MAX_SIZE 100
7
8  // Structure for stack
9  typedef struct {
10     char arr[MAX_SIZE];
11     int top;
12 } Stack;
13
14 // Function to initialise stack
15 void initialise(Stack* stack) {
16     stack->top = -1;
17 }
18
19 // Function to check if stack is empty
20 bool isEmpty(Stack* stack) {
21     return stack->top == -1;
22 }
23
24 // Function to push an element onto the stack
25 void push(Stack* stack, char character) {
26     stack->arr[++stack->top] = character;
27 }
28
29 // Function to pop an element from the stack
30 char pop(Stack* stack) {
31     if (!isEmpty(stack)) {
32         return stack->arr[stack->top--];
33     }
34     return '\0';
35 }
36
37 // Function to check balanced parentheses
38 bool isBalanced(char* str, char* parentheses) {
39     Stack stack;
40     initialise(&stack);
41     for (int i = 0; i < strlen(str); i++) {
42         char character = str[i];
43         if (character == '(' || character == '[' || character == '{') {
44             push(&stack, character); // Push opening parentheses onto the stack
45         } else if (character == ')' || character == ']' || character == '}') {
46             char popped = pop(&stack);
47             if (popped != '(' && character == ')') ||
48                 (popped != '[' && character == ']') ||
49                 (popped != '{' && character == '}') {
50                 return false; // If the opening and closing parentheses don't match, it's unbalanced
51             }
52         }
53     }
54     return !isEmpty(&stack); // If the stack is empty at the end, it's balanced
55 }
56
57 int main() {
58     char parentheses[] = "([{}])";
59     printf("Test 1: %s is %s\n", parentheses, isBalanced(parentheses) ? "Balanced" : "Not Balanced");
60
61     parentheses = "([{}])";
62     printf("Test 2: %s is %s\n", parentheses, isBalanced(parentheses) ? "Balanced" : "Not Balanced");
63
64     parentheses = "([{}])";
65     printf("Test 3: %s is %s\n", parentheses, isBalanced(parentheses) ? "Balanced" : "Not Balanced");
66
67     parentheses = "([{}])";
68     printf("Test 4: %s is %s\n", parentheses, isBalanced(parentheses) ? "Balanced" : "Not Balanced");
69
70     return 0;
71 }

```

```

stack1.h X stack1.c X test.c X
28 Stack stack;
29 initialise(&stack);
40 int len = strlen(parentheses);
41
42 for (int i = 0; i < len; i++) {
43     char character = parentheses[i];
44     if (character == '(' || character == '[' || character == '{') {
45         push(&stack, character); // Push opening parentheses onto the stack
46     } else if (character == ')' || character == ']' || character == '}') {
47         char popped = pop(&stack);
48         if (popped != '(' && character == ')') ||
49             (popped != '[' && character == ']') ||
50             (popped != '{' && character == '}') {
51             return false; // If the opening and closing parentheses don't match, it's unbalanced
52         }
53     }
54 }
55 return !isEmpty(&stack); // If the stack is empty at the end, it's balanced
56
57 int main() {
58     char parentheses[] = "([{}])";
59     printf("Test 1: %s is %s\n", parentheses, isBalanced(parentheses) ? "Balanced" : "Not Balanced");
60
61     parentheses = "([{}])";
62     printf("Test 2: %s is %s\n", parentheses, isBalanced(parentheses) ? "Balanced" : "Not Balanced");
63
64     parentheses = "([{}])";
65     printf("Test 3: %s is %s\n", parentheses, isBalanced(parentheses) ? "Balanced" : "Not Balanced");
66
67     parentheses = "([{}])";
68     printf("Test 4: %s is %s\n", parentheses, isBalanced(parentheses) ? "Balanced" : "Not Balanced");
69
70     return 0;
71 }

```

"C:\Users\Hassan\Desktop\Codes\C & C++\sheet1\bin\Debug\sheet1.exe"

Process returned -1073741819 (0xC0000005) execution time : 2.640 s
Press any key to continue.

13)

main.c [sheet2] - Code::Blocks 20.03

File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help

<global> main(): int

Management

Projects Files FSymbc

Workspace

sheet2

Sources

main.c

queue1.c

queue1.h

queue1.h

```

1 #include <stdio.h>
2 #include "queue1.h"
3 int main()
4 {
5     Type item;
6     int counter=0;
7     Queue section_code;
8     createQueue(&section_code);
9     Queue group_code;
10    createQueue(&group_code);
11    Queue new_queue;
12    createQueue(&new_queue);
13    for(int i=0 ; i<MAX && !isQueueFull(section_code); i++){
14        enqueue(++counter , &section_code);
15    }
16    counter=0;
17    for(int i=0 ; i<MAX && !isQueueFull(group_code); i++){
18        enqueue(counter+=10 , &group_code);
19    }
20    counter=0;
21    for(int i=0 ; i<MAX && !isQueueFull(new_queue); i++){
22        enqueue((dequeue(&section_code))*10+dequeue(&group_code) , &new_queue);
23    }
24    content(new_queue);
25 }

```

Process returned 0 (0x0) execution time : 0.016 s
Press any key to continue.

C:\Users\Hassan\Desktop\Codes\C & C++\DS_Sheets\sheet2\sheet2\main.c C/C++ Windows (CR+LF) WINDOWS-1252 Line 21, Col 54, Pos 622 Insert Read/Write default ENG 9:36 PM 10/28/2023

14)

test.c [sheet1] - Code::Blocks 20.03

File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help

<global>

Management

Projects Files FSymbc

Workspace

sheet1

Sources

stack1.c

stack1.h

test.c

```

1 #include <stdio.h>
2 #include "stack1.h"
3 int main()
4 {
5     type item;
6     int counter=0;
7     Stack group_id;
8     createStack(&group_id);
9     Stack section_code;
10    createStack(&section_code);
11    Stack group_code;
12    createStack(&group_code);
13    for(int i=0 ; i<MAX && !isFull(group_id); i++){
14        push(counter+=10 , &group_id);
15    }
16    content(group_id);
17    printf("-----\n");
18    for(int i=0 ; i<MAX && !isFull(section_code); i++){
19        push(pop(&group_id)/10 , &section_code);
20    }
21    content(section_code);
22    printf("-----\n");
23    for(int i=0 ; i<MAX && !isFull(group_code); i++){
24        push(pop(&section_code)*10%10 , &group_code);
25    }
26    content(group_code);
27 }

```

Process returned 0 (0x0) execution time : 0.016 s
Press any key to continue.

C:\Users\Hassan\Desktop\Codes\C & C++\DS_Sheets\sheet1\test.c C/C++ Windows (CR+LF) WINDOWS-1252 Line 2, Col 19, Pos 37 Insert Read/Write default ENG 11:39 PM 10/28/2023