CCC 204 Data Structures and Algorithms LABORATORY REPORT : LAB 8# - Stacks & Queues

Samano, Rajel Johann
Information Technology Program, CABECS
Colegio San Agustin – Bacolod

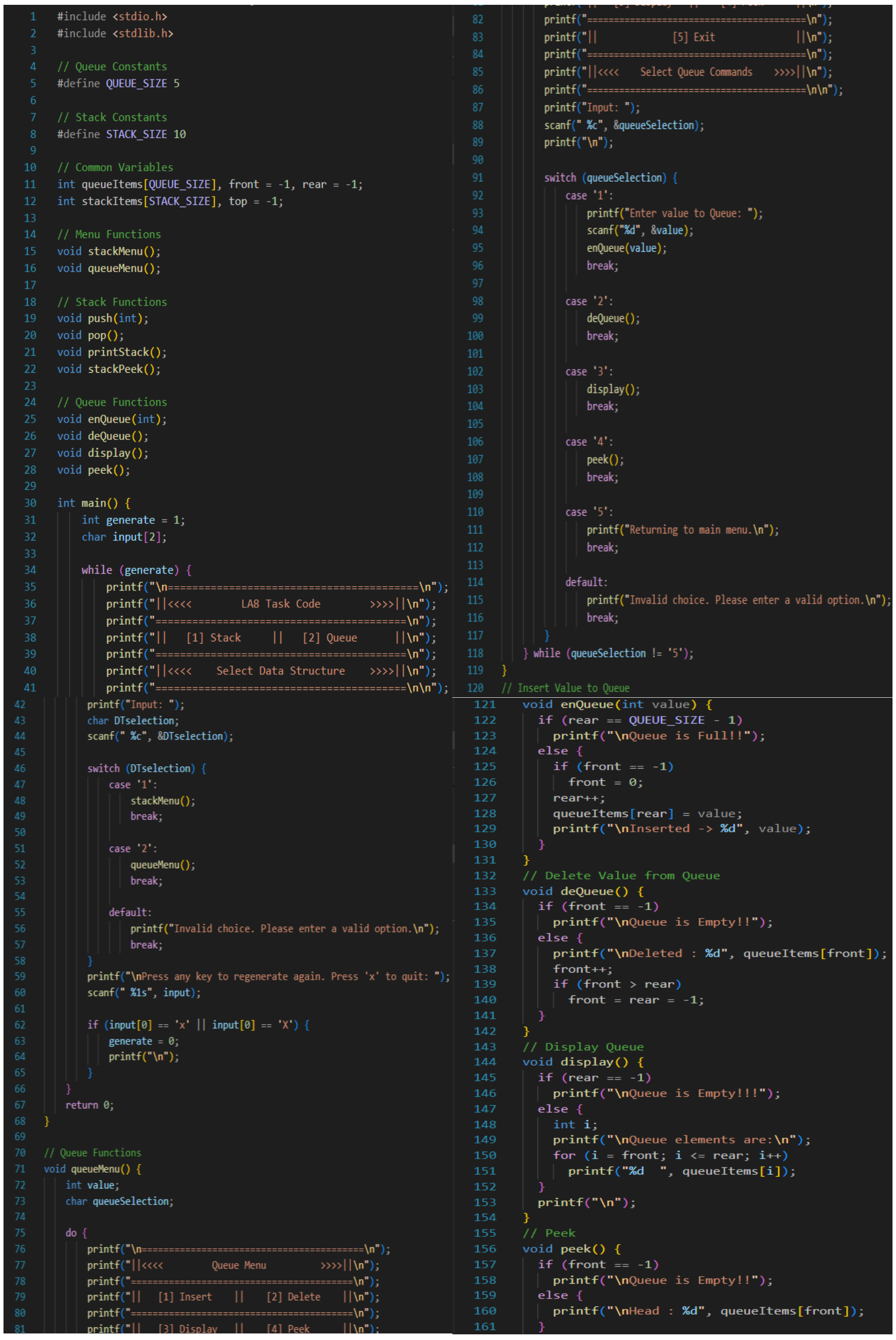
I. INTRODUCTION

For Laboratory Activity Number Seven is to follow the objectives of the exercises and answer questions. The Three main objectives as follows:

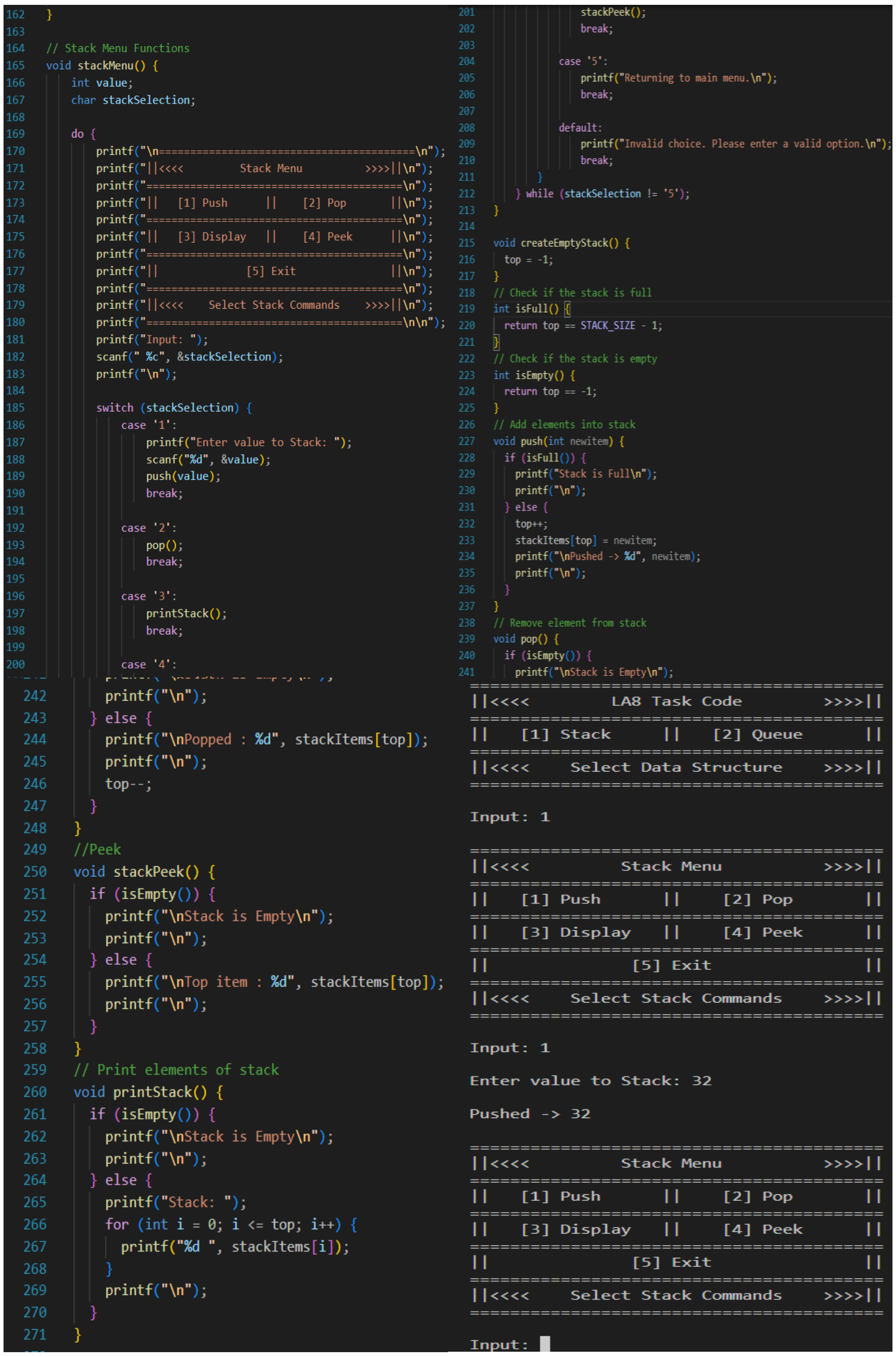
- **Define stack and queue structures**
- **Recognize characteristics and applications of stacks and queue**
- **Observe and analyze stack and queue operation**

II. IMPLEMENTATION / APPROACH

Figure 1-5. LA8_codeTasks.c with output



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

// Queue Constants
#define QUEUE_SIZE 5

// Stack Constants
#define STACK_SIZE 10

// Common Variables
int queueItems[QUEUE_SIZE], front = -1, rear = -1;
int stackItems[STACK_SIZE], top = -1;

// Menu Functions
void stackMenu();
void queueMenu();

// Stack Functions
void push(int);
void pop();
void printStack();
void stackPeek();

// Queue Functions
void enQueue(int);
void deQueue();
void display();
void peek();

int main() {
    int generate = 1;
    char input[2];

    while (generate) {
        printf("\n========================================\n");
        printf("||<<<<        LA8 Task Code        >>>>||\n");
        printf("========================================\n");
        printf("||   [1] Stack     ||   [2] Queue      ||\n");
        printf("========================================\n");
        printf("||<<<<    Select Data Structure    >>>>||\n");
        printf("========================================\n\n");
        printf("Input: ");
        char DTselection;
        scanf(" %c", &DTselection);

        switch (DTselection) {
            case '1':
                stackMenu();
                break;

            case '2':
                queueMenu();
                break;

            default:
                printf("Invalid choice. Please enter a valid option.\n");
                break;
        }
        printf("\nPress any key to regenerate again. Press 'x' to quit: ");
        scanf(" %1s", input);

        if (input[0] == 'x' || input[0] == 'X') {
            generate = 0;
            printf("\n");
        }
    }
    return 0;
}

// Queue Functions
void queueMenu() {
    int value;
    char queueSelection;

    do {
        printf("\n======================================\n");
        printf("||<<<<        Queue Menu        >>>>||\n");
        printf("======================================\n");
        printf("||   [1] Insert    ||    [2] Delete   ||\n");
        printf("======================================\n");
        printf("||   [3] Display   ||    [4] Peek     ||\n");
        printf("======================================\n");
        printf("||             [5] Exit              ||\n");
        printf("======================================\n");
        printf("||<<<<    Select Queue Commands    >>>>||\n");
        printf("======================================\n\n");
        printf("Input: ");
        scanf(" %c", &queueSelection);
        printf("\n");

        switch (queueSelection) {
            case '1':
                printf("Enter value to Queue: ");
                scanf("%d", &value);
                enQueue(value);
                break;

            case '2':
                deQueue();
                break;

            case '3':
                display();
                break;

            case '4':
                peek();
                break;

            case '5':
                printf("Returning to main menu.\n");
                break;

            default:
                printf("Invalid choice. Please enter a valid option.\n");
                break;
        }
    } while (queueSelection != '5');
}
// Insert Value to Queue
void enQueue(int value) {
  if (rear == QUEUE_SIZE - 1)
    printf("\nQueue is Full!!");
  else {
    if (front == -1)
      front = 0;
    rear++;
    queueItems[rear] = value;
    printf("\nInserted -> %d", value);
  }
}
// Delete Value from Queue
void deQueue() {
  if (front == -1)
    printf("\nQueue is Empty!!");
  else {
    printf("\nDeleted : %d", queueItems[front]);
    front++;
    if (front > rear)
      front = rear = -1;
  }
}
// Display Queue
void display() {
  if (rear == -1)
    printf("\nQueue is Empty!!!");
  else {
    int i;
    printf("\nQueue elements are:\n");
    for (i = front; i <= rear; i++)
      printf("%d  ", queueItems[i]);
  }
  printf("\n");
}
// Peek
void peek() {
  if (front == -1)
    printf("\nQueue is Empty!!");
  else {
    printf("\nHead : %d", queueItems[front]);
  }
```



```c
}

// Stack Menu Functions
void stackMenu() {
    int value;
    char stackSelection;

    do {
        printf("\n========================================\n");
        printf("||<<<<         Stack Menu          >>>>||\n");
        printf("========================================\n");
        printf("||   [1] Push      ||    [2] Pop       ||\n");
        printf("========================================\n");
        printf("||   [3] Display   ||    [4] Peek      ||\n");
        printf("========================================\n");
        printf("||                [5] Exit             ||\n");
        printf("========================================\n");
        printf("||<<<<    Select Stack Commands    >>>>||\n");
        printf("========================================\n\n");
        printf("Input: ");
        scanf(" %c", &stackSelection);
        printf("\n");

        switch (stackSelection) {
            case '1':
                printf("Enter value to Stack: ");
                scanf("%d", &value);
                push(value);
                break;

            case '2':
                pop();
                break;

            case '3':
                printStack();
                break;

            case '4':
                stackPeek();
                break;

            case '5':
                printf("Returning to main menu.\n");
                break;

            default:
                printf("Invalid choice. Please enter a valid option.\n");
                break;
        }
    } while (stackSelection != '5');
}

void createEmptyStack() {
  top = -1;
}
// Check if the stack is full
int isFull() {
  return top == STACK_SIZE - 1;
}
// Check if the stack is empty
int isEmpty() {
  return top == -1;
}
// Add elements into stack
void push(int newitem) {
  if (isFull()) {
    printf("Stack is Full\n");
    printf("\n");
  } else {
    top++;
    stackItems[top] = newitem;
    printf("\nPushed -> %d", newitem);
    printf("\n");
  }
}
// Remove element from stack
void pop() {
  if (isEmpty()) {
    printf("\nStack is Empty\n");
    printf("\n");
  } else {
    printf("\nPopped : %d", stackItems[top]);
    printf("\n");
    top--;
  }
}
//Peek
void stackPeek() {
  if (isEmpty()) {
    printf("\nStack is Empty\n");
    printf("\n");
  } else {
    printf("\nTop item : %d", stackItems[top]);
    printf("\n");
  }
}
// Print elements of stack
void printStack() {
  if (isEmpty()) {
    printf("\nStack is Empty\n");
    printf("\n");
  } else {
    printf("Stack: ");
    for (int i = 0; i <= top; i++) {
      printf("%d ", stackItems[i]);
    }
    printf("\n");
  }
}
```

```
========================================
||<<<<        LA8 Task Code        >>>>||
========================================
||   [1] Stack      ||   [2] Queue     ||
========================================
||<<<<   Select Data Structure     >>>>||
========================================

Input: 1

========================================
||<<<<         Stack Menu          >>>>||
========================================
||   [1] Push      ||    [2] Pop       ||
========================================
||   [3] Display   ||    [4] Peek      ||
========================================
||                [5] Exit             ||
========================================
||<<<<    Select Stack Commands    >>>>||
========================================

Input: 1

Enter value to Stack: 32

Pushed -> 32

========================================
||<<<<         Stack Menu          >>>>||
========================================
||   [1] Push      ||    [2] Pop       ||
========================================
||   [3] Display   ||    [4] Peek      ||
========================================
||                [5] Exit             ||
========================================
||<<<<    Select Stack Commands    >>>>||
========================================

Input:
```

My approach towards the problem was first to follow the links that was given to us by our teacher as those links have codes for stacks and queues. That are already working what we the students are supposed to do was to make a menu driven interface which was kind of easy for me as I have been doing a menu driven interface for my previous lab works what I did was copy all the functions from both stacks and queues as it would be less of a hassle to make new functions I remove everything except the condition on the functions createEmptyStack, isFull, and isEmpty after was to return those conditions. For the menu part I just copied the design from my LA7_codeTasks code and have the user select from two data structures which are stacks and queues and when selecting one of those two they are brought to a menu for each of them respectively it is a do while loop which would run the loop again until you press five and then returned to exit the code or generate another.

III. EXPERIMENTAL FINDINGS / DISCUSSIONS

What I found through this is that stacks need a lot more functions as it needs createEmptyStack, isEmpty, and isFull as I tried to find other codes due to me removing the pointers I also found that it can ran fine without struct which I find weird why this one example has a struct while the others didn't have one.

IV. CONCLUSIONS

To conclude this laboratory activity, I could say that I somewhat have a grasp on what I am doing with stacks and queues moving forward from this point as it is simple and I can identify how which is which like how queues is different from stacks as queues are a (FIFO) or first in first out and for stacks it would be (LIFO) or last in first out.

V. References:

https://www.digitalocean.com/community/tutorials/stack-in-c

https://www.tutorialspoint.com/data_structures_algorithms/stack_program_in_c.htm