**BLG 233E**

**Data Structures and Laboratory**

**CRN: 11692**

**3rd Homework**

**Lecturer’s Name: Gülşen Eryiğit**

**Student’s Name: Pelin Hakverir**

**Student’s Number: 150140031**

**Delivery Date: 21.12.2015**

**Introduction**

As stated in description of the homework, it is required from us to write a fortune-telling application with a recursion function to shuffle the cards of the deck.

**Environment**

The application is written on Windows 8.1 with Visual Studio 2013.

There are some loops for traversing in the queues and stacks. The application also uses some specific functions like freopen for writing the outputs on the output.exe file and function(2D-Array[x]+y) for converting 2-D array to a pointer and to send a pointer to the function.

**Used Structures**

The main structure is:

**struct CardFeatures{**

**char \*cardNumber;**

**char cardType;**

**int cardValue;**

**};**

This structure stores the data of the cards. The other structures are created by using this structure as a base:

**struct CreateDeck{ //this struct is created to be used in the main function while controlling the cards if they match**

**CardFeatures \*card;**

**CreateDeck \*next;**

**};**

**struct DeckCards{ //this struct is used to store the cards of the deck in shuffled order**

**CardFeatures \*card;**

**DeckCards \*next;**

**};**

**struct TempCardPlace{ //this struct is used to store the cards which does not match**

**CardFeatures \*card;**

**TempCardPlace \*next;**

**};**

Also there are two more structs for storing the functions of the queue and stack operations:

**struct Stack{ //this struct is used to store the stack functions**

**TempCardPlace \*head;**

**void create();**

**void close();**

**void push(CardFeatures \*);**

**CardFeatures\* pop();**

**bool isempty();**

**};**

**struct Queue{ //this struct is used to store the queue functions**

**DeckCards \*front;**

**DeckCards \*back;**

**void create();**

**void close();**

**void enqueue(CardFeatures\*);**

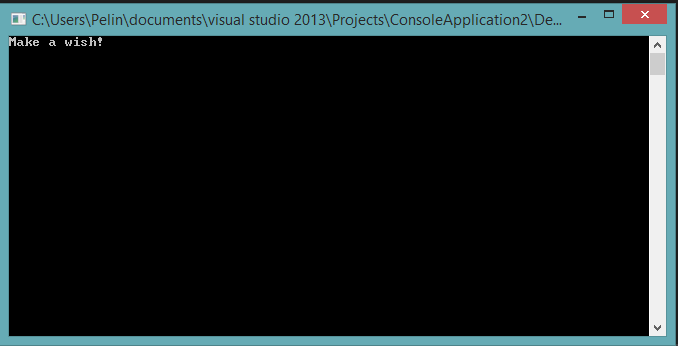
**CardFeatures\* dequeue();**

**bool isempty();**

**};**

**Console Output**

The application have been successfully written and tested on proper windows and linux environments. The screenshot of the console is below:



The program hasn’t ended because I used a getchar and at the very moment I push the enter button, console closes. It is an extremely normal behaviour, program works very well.

**Pseudocode of the Shuffle Function**

**Code:**

**void shuffle(CardFeatures DeckCard[][13]){**

**if (counter < 52){ //I used 52 for storage, it has no effect on swap part (it has about swapping numbers, but it won't be a problem for this part if it was 40 or 60)**

**swap(DeckCard[rand() % 4][rand() % 13], DeckCard[rand() % 4][rand() % 13]); //swap random cards**

**counter++;**

**shuffle(DeckCard); //shuffle other cards**

**}**

**if (counter13 == 13 && counter!=0){ //if a part is passed**

**counter4++; //pass to another type**

**counter13 = 0; //come to the beginning of the new type**

**}**

**if (counter13 < 13 && counter!=0){ //if the type hasn't ended**

**Deck.enqueue(DeckCard[counter4] + counter13); //store it in the deck queue**

**counter13++;**

**}**

**counter--;**

**}**

**Pseudocode:**

**shuffle the given deck of cards**

**if the counter is smaller than 52**

**swap two random cards**

**increase the counter by 1**

**shuffle the deck of cards**

**if the counter13 is not 13 and counter is not 0**

**increase the counter4 by 1**

**set counter13 to 0**

**if the counter13 is smaller than 13 and counter is not 0**

**enqueue the current card**

**increase the counter13 by 1**

**decrease the counter by 1**

**notes:**

**-counter=0 at the beginning**

**-counter13 is used to count the number of the cards in the rows of the deck 2-D array**

**-counter4 is used to pass to another row when its time comes(when the previous row has ended)**

**-counter13 is set to 0 in the “second if” to pass to the beginning of the new row**

**-counter13 is increased to pass to another row of the 2-D array**

**Conclusion**

The application have been successfully written and tested on proper windows and linux environments.