## CSCI 140 Final Submission

## Due Date: 12/09/2021 Late (date and time):\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## Name(s): Nero Li

**Question 1**

Source code below:

/\*  Program: Lab\_Final\_1

    Author: Nero Li

    Class: CSCI 220

    Date: 12/09/2021

    Description:

        Set up MyIntQueue class that supports standard queue operations with int values. The

        class must use two stacks of integers to store data and support stack operations (no

        additional variables in the class). What is the running time for each queue operation?

        You can use your stack class from a PA or one from C++/ Java language. You might

        want to make sure that your implementation would work with the following test case:

        // s is an object of type MyIntQueue

        s.enqueue(5);

        s. enqueue(7);

        x = s.front(); // x is 5

        s. enqueue(1);

        x = s.dequeue(); // x is 5, dequeue() will return a value

        s.enqueue(2);

        x = s.dequeue(); // x is 7

        x = s.dequeue(); // x is 1

    I certify that the code below is my own work.

    Exception(s): N/A

\*/

#include <iostream>

#include <stack>

using namespace std;

class MyIntQueue

{

private:

    stack<int> stk1;

    stack<int> stk2;

public:

    // Running Time: O(1)

    void enqueue(int n)

    {

        stk1.push(n);

    }

    // Running Time: O(n^2)

    int dequeue()

    {

        int n;

        while (!stk1.empty())

        {

            stk2.push(stk1.top());

            stk1.pop();

        }

        n = stk2.top();

        stk2.pop();

        while (!stk2.empty())

        {

            stk1.push(stk2.top());

            stk2.pop();

        }

        return n;

    }

    // Running Time: O(n^2)

    int front()

    {

        int n;

        while (!stk1.empty())

        {

            stk2.push(stk1.top());

            stk1.pop();

        }

        n = stk2.top();

        while (!stk2.empty())

        {

            stk1.push(stk2.top());

            stk2.pop();

        }

        return n;

    }

};

void requirement1()

{

    int x;

    MyIntQueue s;       // s is an object of type MyIntQueue

    s.enqueue(5);

    s.enqueue(7);

    x = s.front();      // x is 5

    cout << x << ' ';

    s.enqueue(1);

    x = s.dequeue();    // x is 5, dequeue() will return a value

    cout << x << ' ';

    s.enqueue(2);

    x = s.dequeue();    // x is 7

    cout << x << ' ';

    x = s.dequeue();    // x is 1

    cout << x << ' ';

    cout << endl;

}

void requirement2()

{

    int a[] = {5, 7, 1, 2, 4};

    int count{5};

    MyIntQueue q1;

    MyIntQueue q2;

    int switcher{1};

    for (int i = 0; i < count; ++i)

    {

        q2.enqueue(a[i]);

    }

    for (int i = 0; i < count; ++i)

    {

        switch (switcher)

        {

        case 1:

            for (int j = 0; j < count - 1 - i; ++j)

            {

                q1.enqueue(q2.dequeue());

            }

            switcher = 2;

            a[i] = q2.dequeue();

            break;

        case 2:

            for (int j = 0; j < count - 1 - i; ++j)

            {

                q2.enqueue(q1.dequeue());

            }

            switcher = 1;

            a[i] = q1.dequeue();

            break;

        default:

            break;

        }

    }

    for (int i = 0; i < count; ++i)

    {

        cout << a[i] << ' ';

    }

    cout << endl;

}

int main()

{

    requirement1();

    requirement2();

    cout << "Author: Nero Li\n";

    return 0;

}

Input/output below:

5 5 7 1

4 2 1 7 5

Author: Nero Li

**Question 2 - skipped**

**Question 3**

Source code below:

/\*  Program: Lab\_Final\_3

    Author: Nero Li

    Class: CSCI 220

    Date: 12/09/2021

    Description:

        Implement a C++ class template or Java class generic MyArrayList that uses an array to

        include the following operations: insertRear(e), removeFront(), elementAt(i), empty(),

        and cap(). You must set up a dynamic array with room for up to 10 elements and you

        can assume that there is at least one element when removeFront() is called. There is a

        penalty of 5 points if it is not a C++ class template or Java class generic. What is the

        running time for each operation?

    I certify that the code below is my own work.

    Exception(s): N/A

\*/

#include <iostream>

using namespace std;

template <typename T>

class MyArrayList

{

private:

    T \*a;

    int cap;

    int amount;

protected:

    // Running Time: O(n) when array need to expand

    void expandArray()

    {

        if (amount >= cap)

        {

            cap \*= 2;

            T \*b = a;

            a = new T[cap];

            for (int i = 0; i < amount; ++i)

            {

                a[i] = b[i];

            }

            delete [] b;

        }

    }

public:

    MyArrayList()

    {

        cap = 2;

        amount = 0;

        a = new T[cap];

    };

    // Running Time: O(1)

    void insertRear(T e)

    {

        a[amount++] = e;

        expandArray();

    }

    // Running Time: O(n)

    void removeFront()

    {

        for (int i = 0; i < amount; ++i)

        {

            a[i] = a[i + 1];

        }

        --amount;

    }

    // Running Time: O(1)

    T elementAt(int i)

    {

        return a[i];

    }

    // Running Time: O(1)

    bool empty()

    {

        return amount == 0;

    }

    // Running Time: O(1)

    int size()

    {

        return amount;

    }

    // Running Time: O(n)

    void print()

    {

        for (int i = 0; i < amount; i++)

        {

            cout << a[i] << endl;

        }

    }

    // Running Time: O(n)

    void insertFront(T e)

    {

        for (int i = amount - 1; i > 0; --i)

        {

            a[i] = a[i - 1];

        }

        ++amount;

        expandArray();

        a[0] = e;

    }

    // Running Time: O(1)

    void removeRear()

    {

        --amount;

    }

};

void requirement1()

{

    MyArrayList<int> test;

    cout << test.empty() << ' ';        // Output: 1

    test.insertRear(3);                 // List: 3

    test.insertRear(7);                 // List: 3 7

    cout << test.elementAt(0) << ' ';   // Output: 3

    test.insertRear(4);                 // List: 3 7 4

    test.removeFront();                 // List: 7 4

    test.insertRear(9);                 // List: 7 4 9

    cout << test.elementAt(2) << ' ';   // Output: 9

    test.insertRear(2);                 // List: 7 4 9 2

    test.insertRear(8);                 // List: 7 4 9 2 8

    test.insertRear(6);                 // List: 7 4 9 2 8 6

    test.removeFront();                 // List: 4 9 2 8 6

    cout << endl;

    for (int i = 0; i < test.size(); ++i)

    {

        cout << test.elementAt(i) << ' ';

    }

    cout << endl;

}

void requirement2()

{

    MyArrayList<string> s;

    s.insertFront("CSCI 140");

    s.insertRear("CSCI 145");

    s.insertRear("CSCI 220");

    s.insertFront("CSCI 110");

    cout << "size(): " << s.size() << endl;

    cout << "empty(): " << (s.empty() ? "True" : "False") << endl;

    cout << "elementAt(2): " << s.elementAt(2) << endl;

    s.removeFront();

    s.removeRear();

    s.print();

}

int main()

{

    requirement1();

    requirement2();

    cout << "Author: Nero Li\n";

    return 0;

}

Input/output below:

1 3 9

4 9 2 8 6

size(): 4

empty(): False

elementAt(2): CSCI 145

CSCI 140

CSCI 145

Author: Nero Li