**Assignment 3**

1-Generate Binary Numbers

* Generate the first n binary numbers using a queue.
* Input: n = 5
* Output: 1, 10, 11, 100, 101

**Code:**

#*include* <iostream>

#*include* <string>

using namespace std;

class MyQueue {

private:

    string\* data;

    int frontIndex;

    int backIndex;

    int size;

    int capacity;

public:

*MyQueue*(int cap) {

        data = new string[cap];

        frontIndex = 0;

        backIndex = -1;

        size = 0;

        capacity = cap;

    }

*~MyQueue*() {

        delete[] data;

    }

    bool *isEmpty*() {

*return* size == 0;

    }

    bool *isFull*() {

*return* size == capacity;

    }

    void *push*(*const* string*&* value) {

*if* (!*isFull*()) {

            backIndex = (backIndex + 1) % capacity;

            data[backIndex] *=* value;

            size++;

        } *else* {

            cout *<<* "Queue is full. Cannot add more elements." *<<* *endl*;

        }

    }

    string *front*() {

*if* (*isEmpty*()) {

*return* "";

        }

*return* data[frontIndex];

    }

    void *pop*() {

*if* (!*isEmpty*()) {

            frontIndex = (frontIndex + 1) % capacity;

            size--;

        }

    }

    int *getSize*() {

*return* size;

    }

};

string*\** *generateBinaryNumbers*(int n) {

*if* (n <= 0) {

*return* nullptr;

    }

    string\* result = new string[n];

    MyQueue *q*(n \* 2);

    string initial = "1";

    q.*push*(initial);

*for* (int i = 0; i < n; ++i) {

        string current = q.*front*();

        q.*pop*();

        result[i] *=* current;

        string zero = current *+* "0";

        string one = current *+* "1";

        q.*push*(zero);

        q.*push*(one);

    }

*return* result;

}

int *main*() {

    int n;

    cout *<<* "Enter the number of binary numbers to generate (positive integer only): ";

    cin *>>* n;

*if* (n <= 0) {

        cout *<<* "Invalid input. Please enter a positive integer." *<<* *endl*;

*return* 1;

    }

    string\* binaryNumbers = *generateBinaryNumbers*(n);

    cout *<<* "The first " *<<* n *<<* " binary numbers are: ";

*for* (int i = 0; i < n; ++i) {

        cout *<<* binaryNumbers[i] *<<* " ";

    }

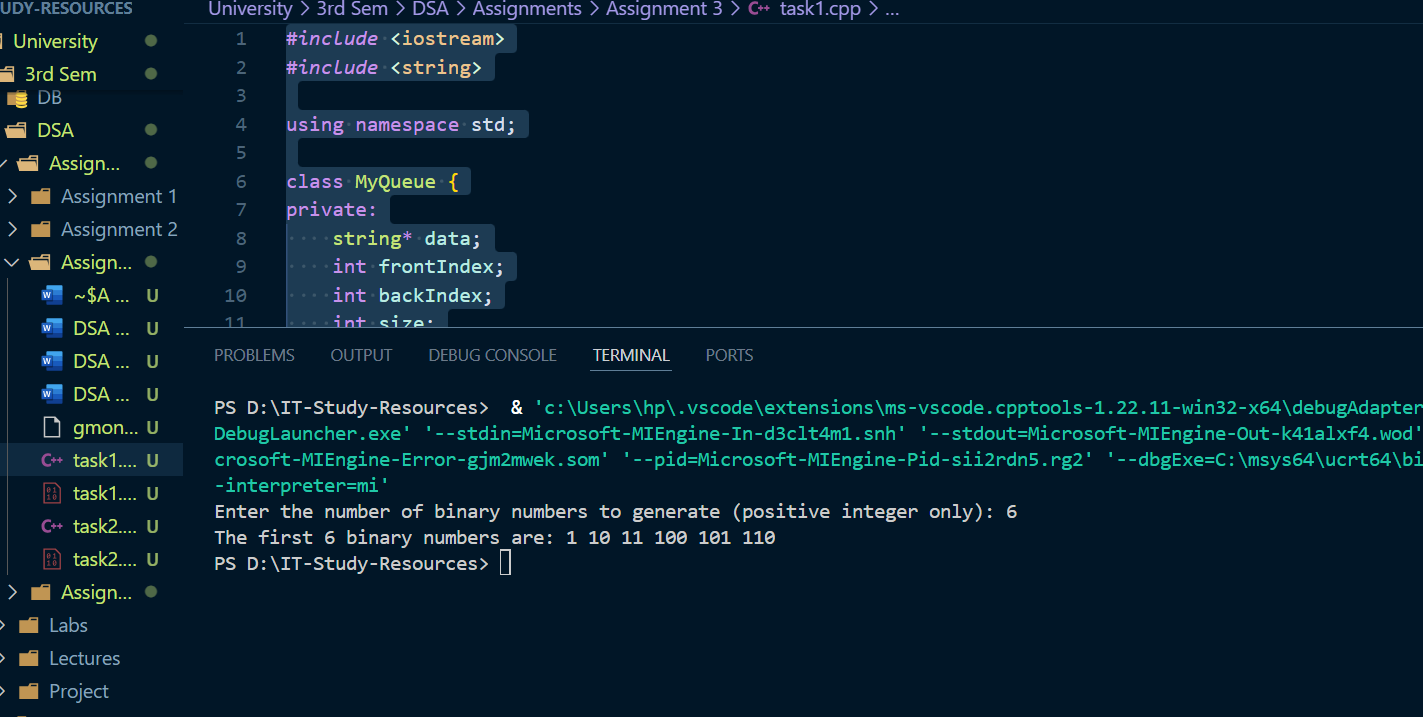
    cout *<<* *endl*;

    delete[] binaryNumbers;

*return* 0;

}

**Output:**

****

2- Interleave the First Half and Second Half of a Queue

* For example: Input queue: 1, 2, 3, 4, 5, 6, Output: 1, 4, 2, 5, 3, 6.

**Code:**

#*include* <iostream>

#*include* <string>

using namespace std;

class MyQueue {

private:

    int\* data;

    int frontIndex;

    int backIndex;

    int size;

    int capacity;

public:

*MyQueue*(int cap) {

        data = new int[cap];

        frontIndex = 0;

        backIndex = -1;

        size = 0;

        capacity = cap;

    }

*~MyQueue*() {

        delete[] data;

    }

    bool *isEmpty*() {

*return* size == 0;

    }

    bool *isFull*() {

*return* size == capacity;

    }

    void *enqueue*(*const* int*&* value) {

*if* (!*isFull*()) {

            backIndex = (backIndex + 1) % capacity;

            data[backIndex] = value;

            size++;

        } *else* {

            cout *<<* "Queue is full. Cannot add more elements." *<<* *endl*;

        }

    }

    int *dequeue*() {

*if* (*isEmpty*()) {

            cout *<<* "Queue is empty. Cannot dequeue." *<<* *endl*;

*return* -1; *// Indicating error*

        }

        int frontValue = data[frontIndex];

        frontIndex = (frontIndex + 1) % capacity;

        size--;

*return* frontValue;

    }

    int *getSize*() {

*return* size;

    }

    void *print*() {

*if* (*isEmpty*()) {

            cout *<<* "Queue is empty." *<<* *endl*;

*return*;

        }

*for* (int i = 0; i < size; ++i) {

            cout *<<* data[(frontIndex + i) % capacity] *<<* " ";

        }

        cout *<<* *endl*;

    }

};

void *interleaveQueue*(MyQueue*&* q) {

    int n = q.*getSize*();

*if* (n % 2 != 0) {

        cout *<<* "Please provide an even number of elements." *<<* *endl*;

*return*;

    }

    MyQueue *tempQueue*(n / 2);

    int halfSize = n / 2;

*for* (int i = 0; i < halfSize; ++i) {

        tempQueue.*enqueue*(q.*dequeue*());

    }

*for* (int i = 0; i < halfSize; ++i) {

        q.*enqueue*(tempQueue.*dequeue*());

        q.*enqueue*(q.*dequeue*());

    }

}

int *main*() {

    MyQueue *q*(10);

    cout *<<* "Enter 6 integers: ";

*for* (int i = 0; i < 6; ++i) {

        int num;

        cin *>>* num;

        q.*enqueue*(num);

    }

    cout *<<* "Queue before interleaving: ";

    q.*print*();

*interleaveQueue*(q);

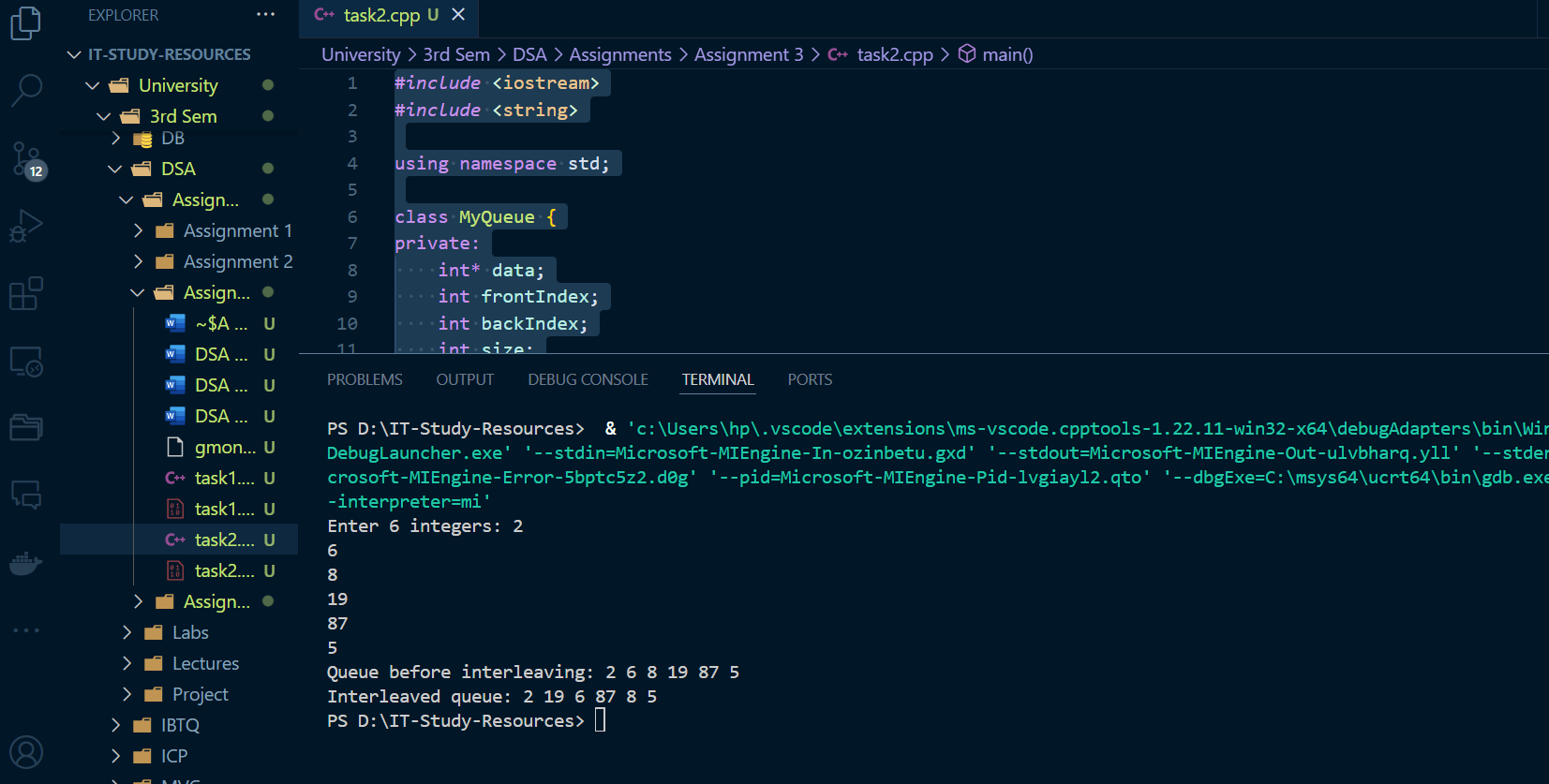
    cout *<<* "Interleaved queue: ";

    q.*print*();

*return* 0;

}

**Output:**

****