C++ Programming
Structures Practice

Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)

Practice: Our own queue

- Define a class, name it queue. It should internally have an array and support following operations
  - void add_end(int value): add to the end of current array
  - void add_front(int value): add to the front of this array
  - int remove_front(): remove the front value and remove it. Return the value
  - void print(): print the array

Practice: Our own queue

```
int main() {
    queue my_queue;

    my_queue.add_end(10);
    my_queue.add_end(20);
    my_queue.add_end(30);
    my_queue.print();

    my_queue.add_front(1);
    my_queue.add_front(4);
    my_queue.print();

    cout<<my_queue.remove_front();

    return 0;
}
```

```
<terminated> ztem
10 20 30
4 1 10 20 30
4
```

Practice: Our own queue

14_7.cpp

```cpp
#include<iostream>
using namespace std;

struct queue {
    int arr[100];
    int len;

    queue() {
        len = 0;
    }

    void add_end(int value) {
        arr[len++] = value;
    }
    void add_front(int value) {
        // Shift right
        for (int i = len-1; i >= 0; --i)
            arr[i+1] = arr[i];
        arr[0] = value;
        len++;
    }
    int remove_front() {
        int ret = arr[0];
        // Shift left
        for (int i = 1; i < len; ++i)
            arr[i-1] = arr[i];
        --len;
        return ret;
    }

    void print() {
        for (int i = 0; i < len; ++i)
            cout<<arr[i]<<" ";
        cout<<"\n";
    }
};
```

Practice - Hospital System

- Let’s rewrite again the last hospital system
- Give yourself ‘good trial’ in thinking how to re-write

Practice - Hospital System

- Let’s create hospital_queue, like the previous queue
  - Variables
  - string names[MAX_QUEUE];
  - int status[MAX_QUEUE];
  - int len;
  - int spec;
  - Provide same functionalities
- Let’s create hospital_system
  - hospital_queue queues[MAX_SPECIALIZATION];
  - Add the methods inside it using the hospital_queue change

Practice - Hospital System - Big Picture

```
// Global variables
const int MAX_SPECIALIZATION = 20;
const int MAX_QUEUE = 5;

struct hospital_queue {
    string names[MAX_QUEUE];
    int status[MAX_QUEUE];
    int len;
    int spec;

    hospital_queue() {..

    hospital_queue(int _spec) {..

    bool add_end(string name, int st) {..

    bool add_front(string name, int st) {..

    void remove_front() {..

    void print() {..
};
```

```
struct hospital_system {
    hospital_queue queues[MAX_SPECIALIZATION];

    hospital_system() {..

    void run() {..

    int menu() {..

    bool add_patient() {..

    void print_patients() {..

    void get_next_patients() {..
};

int main() {
    freopen("c.in", "rt", stdin);
    hospital_system hospital = hospital_system();
    hospital.run();
    return 0;
}
```

Practice - Hospital System

14_8_hospital_v2.cpp c.in

```cpp
#include<iostream>
using namespace std;

// Global variables
const int MAX_SPECIALIZATION = 20;
const int MAX_QUEUE = 5;

struct hospital_queue {
    string names[MAX_QUEUE];
    int status[MAX_QUEUE];
    int len;
    int spec;

    hospital_queue() {
        len = 0;
        spec = -1;
    }

    hospital_queue(int _spec) {
        len = 0;
        spec = _spec;
    }

    bool add_end(string name, int st) {
        if (len == MAX_QUEUE)
            return false;
        names[len] = name, status[len] = st, ++len;
        return true;
    }
```

Practice - Hospital System

```
void remove_front() {
    if (len == 0) {
        cout << "No patients at the moment. Have rest, Dr\n";
        return;
    }
    cout << names[0] << " please go with the Dr\n";

    // Shift left
    for (int i = 1; i < len; ++i) {
        names[i - 1] = names[i];
        status[i - 1] = status[i];
    }
    --len;
}
void print() {
    if (len == 0)           return;
    cout << "There are " << len << " patients in specialization " <
    for (int i = 0; i < len; ++i) {
        cout << names[i] << " ";
        if (status[i])      cout << "urgent\n";
        else                cout << "regular\n";
    }
    cout << "\n";
}
```

Practice - Hospital System

```
struct hospital_system {
    hospital_queue queues[MAX_SPECIALIZATION];

    hospital_system() {
        for (int i = 0; i < MAX_SPECIALIZATION; ++i)
            queues[i] = hospital_queue(i);
    }

    void run() {
        while (true) {
            int choice = menu();

            if (choice == 1)
                add_patient();
            else if (choice == 2)
                print_patients();
            else if (choice == 3)
                get_next_patients();
            else
                break;
        }
    }
```

```
int menu() {
    int choice = -1;
    while (choice == -1) {
        cout << "\nEnter your choice:\n";
        cout << "1) Add new patient\n";
        cout << "2) Print all patients\n";
        cout << "3) Get next patient\n";
        cout << "4) Exit\n";

        cin >> choice;

        if (!(1 <= choice && choice <= 4)) {
            cout << "Invalid choice. Try again\n";
            choice = -1;    // loop keep working
        }
    }
    return choice;
}
```

Practice - Hospital System

```
bool add_patient() {
    int spec, st;
    string name;

    cout << "Enter specialization, name, status: ";
    cin >> spec >> name >> st;

    bool status;
    if (st == 0)
        status = queues[spec].add_end(name, st);
    else
        status = queues[spec].add_front(name, st);

    if (status == false) {
        cout
                << "Sorry we can't add more patients "
                        "for this specialization\n";
        return false;
    }

    return true;
}
```

Practice - Hospital System

```
void print_patients() {
    cout << "*****************************\n";
    for (int spec = 0; spec < MAX_SPECIALIZATION; ++spec)
        queues[spec].print();
}

void get_next_patients() {
    int spec;
    cout << "Enter specialization: ";
    cin >> spec;
    queues[spec].remove_front();
}
```

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."