



**VIT<sup>®</sup>**  
**Vellore Institute of Technology**  
(Deemed to be University under section 3 of UGC Act, 1956)

# **Cycle Management System using Stack and Queue**

## **A J-Component Report**

*Submitted by Team member(s)*

Name	Reg no.	Slot
Vikram Baruah	18BEC0967	C1

Under the guidance of

**Prof. Delhi Babu R**

*October 2020*

## Table of Contents

Serial no.	Title	Page no.
1.	Introduction	3.
	1.1 Abstract	3.
	1.2 Softwares used	3.
	1.3 Working	3.
2.	Code	4.
3.	Input & Output	11.
4.	Results	15.
5.	Conclusion	16.
6.	References	16.

## **I. Introduction**

We are well aware of the fact that the academic blocks in VIT are located quite far away from each other and from both the Men's Hostel and Ladies' Hostel. Classes may start as early as 8 in the morning and go on till 7 in the evening, and without proper means of transportation it may drain the student out because of different classes being held in different blocks.

The only transportation facility available, currently in VIT are Shuttles. The shuttles, however, have a few set of limitations to them. They only cover the major academic blocks and charge a constant rate irrespective of the distance travelled. Also, shuttles are not very reliable as there are only a limited number available and arrive at a location, say Men's Hostel after it has completed one full cycle.

In order to tackle this problem I have proposed a Cycle Management System which provides cycles for rent with rental stops being located throughout the campus. A rate has been set based on the locations travelled to, for eg. A rate of Rs. 15 may be set for renting a cycle from Men's Hostel to SJT and a rate of Rs. 10 for Men's Hostel to TT.

## **I. I. Abstract**

A C++ based file management system has been developed, but unlike other file management systems, stack and queues have been used. The project consists of various rental stops being situated at the most frequently travelled to locations within the campus. At the stands there will be people employed who will handle the transaction of the cycle. The employees will be given access to the authority side whereas the user will have access to the user's side of the program. The travel-system would be available to the student right after the verified account creation, which he/she can use anytime according to his/her need. The profit margins of this project would be high, at minimal user rates, and will also be beneficial to the environment. The initial cost is a little high but the return and profit margin definitely makes it worth the investment.

## **I. II. Softwares used**

Sublime text 3 has been used to write the C++ program and bits/stdc++.h library has been included as it contains all the functions required in order to execute the program.

Why C++?

C++ is easily understood by the device, thus reducing the space complexity and it also allows usage across multiple devices. C++ can be easily embedded into other languages.

## **I. III. Working**

The program is divided into three subprograms which involve the Authority side of the program, the User, and one program containing all the functions used in the Authority and the User sides.

The Authority side is concerned with the booking of the cycles and have access to all the files stored in the database created by the users which include their names, registration numbers, password and balance. These attributes are then used to verify whether the user booking the cycle is legitimate, and if the password and registration numbers are verified, the program goes onto the next stage where the starting point and the destination of the cycle rental is selected and the balance is deducted from the account based on the distance between the two points.

The user side of the program is concerned with the creation of database of all the users willing to access the cycles. The information entered include their names, registration number, password, and initial balance. The user also has the option to add balance later if required.

After the user has reached the destination, he/she is required to enter the cycle number and lock assigned to him/her by the person at the cycle rental stop which he/she travelled from.

## II. Code

### 1. User

```
#include "stack_queue.cpp"
int main()
{
    int ch,i=1;
    string reg;
    while(i!=2)
    {
        cout<<endl<<endl<<"Enter your choice: "<<endl;
        cout<<"1. Add Cycle"<<endl;
        cout<<"2. Add Student"<<endl;
        cout<<"3. Add balance"<<endl;
        cin>>ch;
        switch(ch)
        {
            case 1:
                addCycle();
                break;
            case 2:
                newStudent();
                break;
            case 3:
                cout<<"Enter your registration number: ";
                cin>>reg;
                addBalance();
                break;
            default:
                cout<<"Check your input";

        }
        cout<<endl<<"Do you want to:"<<endl;
        cout<<"1. Continue\n2. Exit\n";
        cin>>i;
        if (i==2){
            createfile();
        }
    }
    return(0);
}
```

## 2. Authority

```
#include "stack_queue.cpp"
using namespace std;
int main() {
    char c[10];
    cout<<"Enter the cycle registration number: ";
    cin>>c;
    bookCycle(c);
}
```

## 3. All functions

### Using stack

```
#include<bits/stdc++.h>
using namespace std;
class student{
public:
    stack<string> name;
    stack<string> regno;
    stack<string> password;
    stack<int> balance;
};
class student_database{
public:
    string name;
    string regno;
    string pass;
    int balance;
};
class cycle
{
public:
    char cyclereg[10];
    int lock;
};

void newStudent(){
    auto start = high_resolution_clock::now();
    student st;
    string n;
    string reg;
    string pass;
    int bal;
    cout<<"Enter your name: ";
    cin>>n;
    cout<<"Enter registration number: ";
    cin>>reg;
    cout<<"Enter password: ";
    cin>>pass;
    cout<<"Enter balance: ";
```

```

    cin>>bal;
    st.name.push(n);
    st.regno.push(reg);
    st.password.push(pass);
    st.balance.push(bal);
    auto stop = high_resolution_clock::now();
    auto duration = duration_cast<microseconds>(stop - start);
    cout << endl << duration.count() <<" us"<<endl;
}
void addBalance(){
    auto start = high_resolution_clock::now();
    student st;
    cout<<"Enter balance: ";
    int a;
    cin>>a;
    int temp=st.balance.top();
    temp+=a;
    st.balance.pop();
    st.balance.push(temp);
    auto stop = high_resolution_clock::now();
    auto duration = duration_cast<microseconds>(stop - start);
    cout << endl << duration.count() <<" us"<<endl;
}
void createfile(){
    auto start = high_resolution_clock::now();
    student st;
    student_database obj;
    while(!st.balance.empty()){
        fstream iofile;
        obj.name=st.name.top();
        obj.regno=st.regno.top();
        obj.pass=st.password.top();
        obj.balance=st.balance.top();
        /*cout<<obj.name<<endl<<obj.regno<<endl;*/
        st.name.pop();
        st.regno.pop();
        st.password.pop();
        st.balance.pop();
        iofile.open(obj.regno,ios::out);
        iofile.write((char *) & obj, sizeof(student_database));
        iofile.close();
    }
    auto stop = high_resolution_clock::now();
    auto duration = duration_cast<microseconds>(stop - start);
    cout << endl << duration.count() <<" us"<<endl;
}
void addCycle()
{
    auto start = high_resolution_clock::now();
    fstream iofile;
    cycle cy;
    cout<<"Enter the cycle registration number: ";
    cin>>cy.cyclereg;
    cout<<"Enter the cycle lock number: ";

```

```

cin>>cy.lock;
iofile.open(cy.cyclereg,ios::out);
iofile.write((char *) & cy, sizeof(cycle));
iofile.close();
cout<<"Cycle added SUCCESSFULLY!!";
auto stop = high_resolution_clock::now();
auto duration = duration_cast<microseconds>(stop - start);
cout << endl << duration.count() <<" us"<<endl;
}
void bookCycle(string s)
{
    auto start = high_resolution_clock::now();
    fstream iofile;
    fstream infile;
    cycle cy;
    student_database obj;
    int a,ch;
    int pass;
    string pass2;
    string reg;
    cout<<"Enter your registration no. ";
    cin>>reg;
    iofile.open(s,ios::out|ios::in);
    infile.open(reg,ios::out|ios::in);
    if(!iofile.fail())
    {
        iofile.read((char *) & cy, sizeof(cycle));
        infile.read((char *) & obj, sizeof(student_database));
        cout<<"Enter the cycle password: ";
        cin>>pass;
        cout<<"Enter your password to continue: ";
        cin>>pass2;
        if(pass==cy.lock&&reg==obj.regno&&pass2==obj.pass)
        {
            cout<<"Choose your option"<<endl<<"1. MH to SJT"<<endl<<"2. MH to
                TT"<<endl<<"3. MH to MB"<<endl<<"4. TT to SJT"<<endl<<"5. TT to
                SMV"<<endl<<"6. MB to SJT\n"<<"7. SMV to SJT\n";
            cin>>ch;
            switch (ch){
                case 1: a=15;
                        break;
                case 2: a=10;
                        break;
                case 3: a=5;
                        break;
                case 4: a=5;
                        break;
                case 5: a=5;
                        break;
                case 6: a=10;
                        break;
                case 7: a=7;
                        break;
            }
        }
    }
}

```

```

        if (obj.balance<a){
            cout<<"Not enough balance";
        }
        else {
            obj.balance=obj.balance-a;
            cout<<"New Balance is: "<<obj.balance<<endl;
            cout<<"Enjoy your ride";
        }
        iofile.seekg(0);
        iofile.write((char *) & obj, sizeof(student_database));
    }
    else
        cout<<"Enter a valid password!!";

    iofile.close();
}
else
    cout<<"Enter a registered Registration Number";
    auto stop = high_resolution_clock::now();
    auto duration = duration_cast<microseconds>(stop - start);
    cout << endl << duration.count() << " us"<<endl;

}

```

## Using queue

```

#include<bits/stdc++.h>
using namespace std;
class student{
public:
    queue<string> name;
    queue<string> regno;
    queue<string> password;
    queue<int> balance;
};
class student_database{
public:
    string name;
    string regno;
    string pass;
    int balance;
};
class cycle
{
    public:
    char cyclereg[10];
    int lock;
};

void newStudent(){
    auto start = high_resolution_clock::now();
    student st;
    string n;
    string reg;

```



```

string pass;
int bal;
cout<<"Enter your name: ";
cin>>n;
cout<<"Enter registration number: ";
cin>>reg;
cout<<"Enter password: ";
cin>>pass;
cout<<"Enter balance: ";
cin>>bal;
st.name.push(n);
st.regno.push(reg);
st.password.push(pass);
st.balance.push(bal);
auto stop = high_resolution_clock::now();
auto duration = duration_cast<microseconds>(stop - start);
cout << endl << duration.count() <<" us"<<endl;
}

void addBalance(){
    auto start = high_resolution_clock::now();
    student st;
    cout<<"Enter balance: ";
    int a;
    cin>>a;
    int temp=st.balance.front();
    temp+=a;
    st.balance.pop();
    st.balance.push(temp);
    auto stop = high_resolution_clock::now();
    auto duration = duration_cast<microseconds>(stop - start);
    cout << endl << duration.count() <<" us"<<endl;
}

void createfile(){
    auto start = high_resolution_clock::now();
    student st;
    student_database obj;
    while(!st.balance.empty()){
        fstream iofile;
        obj.name=st.name.front();
        obj.regno=st.regno.front();
        obj.pass=st.password.front();
        obj.balance=st.balance.front();
        /*cout<<obj.name<<endl<<obj.regno<<endl;*/
        st.name.pop();
        st.regno.pop();
        st.password.pop();
        st.balance.pop();
        iofile.open(obj.regno,ios::out);
        iofile.write((char *) & obj, sizeof(student_database));
        iofile.close();
    }
    auto stop = high_resolution_clock::now();
    auto duration = duration_cast<microseconds>(stop - start);
    cout << endl << duration.count() <<" us"<<endl;
}

```

```

}
void addCycle()
{
    auto start = high_resolution_clock::now();
    fstream iofile;
    cycle cy;
    cout<<"Enter the cycle registration number: ";
    cin>>cy.cyclereg;
    cout<<"Enter the cycle lock number: ";
    cin>>cy.lock;
    iofile.open(cy.cyclereg,ios::out);
    iofile.write((char *) & cy, sizeof(cycle));
    iofile.close();
    cout<<"Cycle added SUCCESSFULLY!!";
    auto stop = high_resolution_clock::now();
    auto duration = duration_cast<microseconds>(stop - start);
    cout << endl << duration.count() <<" us"<<endl;
}
void bookCycle(string s)
{
    auto start = high_resolution_clock::now();
    fstream iofile;
    fstream infile;
    cycle cy;
    student_database obj;
    int a,ch;
    int pass;
    string pass2;
    string reg;
    cout<<"Enter your registration no. ";
    cin>>reg;
    iofile.open(s,ios::out|ios::in);
    infile.open(reg,ios::out|ios::in);
    if(!iofile.fail())
    {
        iofile.read((char *) & cy, sizeof(cycle));
        infile.read((char *) & obj, sizeof(student_database));
        cout<<"Enter the cycle password: ";
        cin>>pass;
        cout<<"Enter your password to continue: ";
        cin>>pass2;
        if(pass==cy.lock&&reg==obj.regno&&pass2==obj.pass)
        {
            cout<<"Choose your option"<<endl<<"1. MH to SJT"<<endl<<"2. MH to TT"<<endl<<"3. MH to MB" <<endl<<"4. TT to SJT"<<endl<<"5. TT to SMV"<<endl<<"6. MB to SJT\n"<<"7. SMV to SJT\n";
            cin>>ch;
            switch (ch){
                case 1: a=15;
                    break;
                case 2: a=10;
                    break;
                case 3: a=5;
                    break;
            }
        }
    }
}

```

```

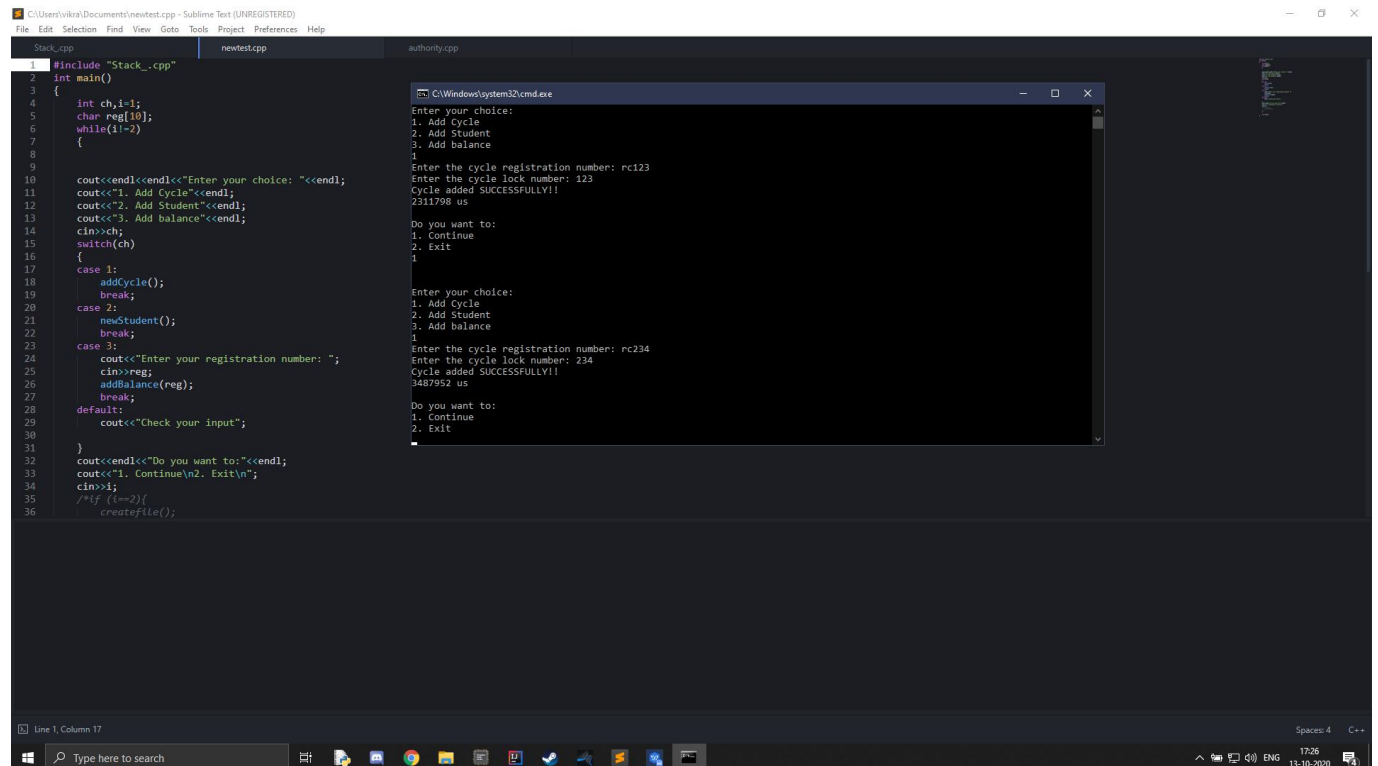
        case 4: a=5;
                break;
        case 5: a=5;
                break;
        case 6: a=10;
                break;
        case 7: a=7;
                break;
    }
    if (obj.balance<a){
        cout<<"Not enough balance";
    }
    else {
        obj.balance=obj.balance-a;
        cout<<"New Balance is: "<<obj.balance<<endl;
        cout<<"Enjoy your ride";
    }
    ifile.seekg(0);
    ifile.write((char *) & obj, sizeof(student_database));
}
else
    cout<<"Enter a valid password!!";

ifile.close();
}
else
    cout<<"Enter a registered Registration Number";
auto stop = high_resolution_clock::now();
auto duration = duration_cast<microseconds>(stop - start);
cout << endl << duration.count() << " us"<<endl;
}

```

### III. Input & Output

#### Adding cycles initially into database



```
1 #include "Stack_.cpp"
2 int main()
3 {
4     int ch,i=1;
5     char reg[10];
6     while(i!=2)
7     {
8
9
10
11         cout<<endl<<"Enter your choice: "<<endl;
12         cout<<"1. Add Cycle"<<endl;
13         cout<<"2. Add Student"<<endl;
14         cout<<"3. Add balance"<<endl;
15         cin>>ch;
16         switch(ch)
17         {
18             case 1:
19                 addCycle();
20                 break;
21             case 2:
22                 newStudent();
23                 break;
24             case 3:
25                 cout<<"Enter your registration number: ";
26                 cin>>reg;
27                 addBalance(reg);
28                 break;
29             default:
30                 cout<<"Check your input";
31         }
32         cout<<endl<<"Do you want to:"<<endl;
33         cout<<"1. Continue\n2. Exit\n";
34         cin>>i;
35         /*if (i==2){
36             createfile();
37         }*/
38     }
39 }
```

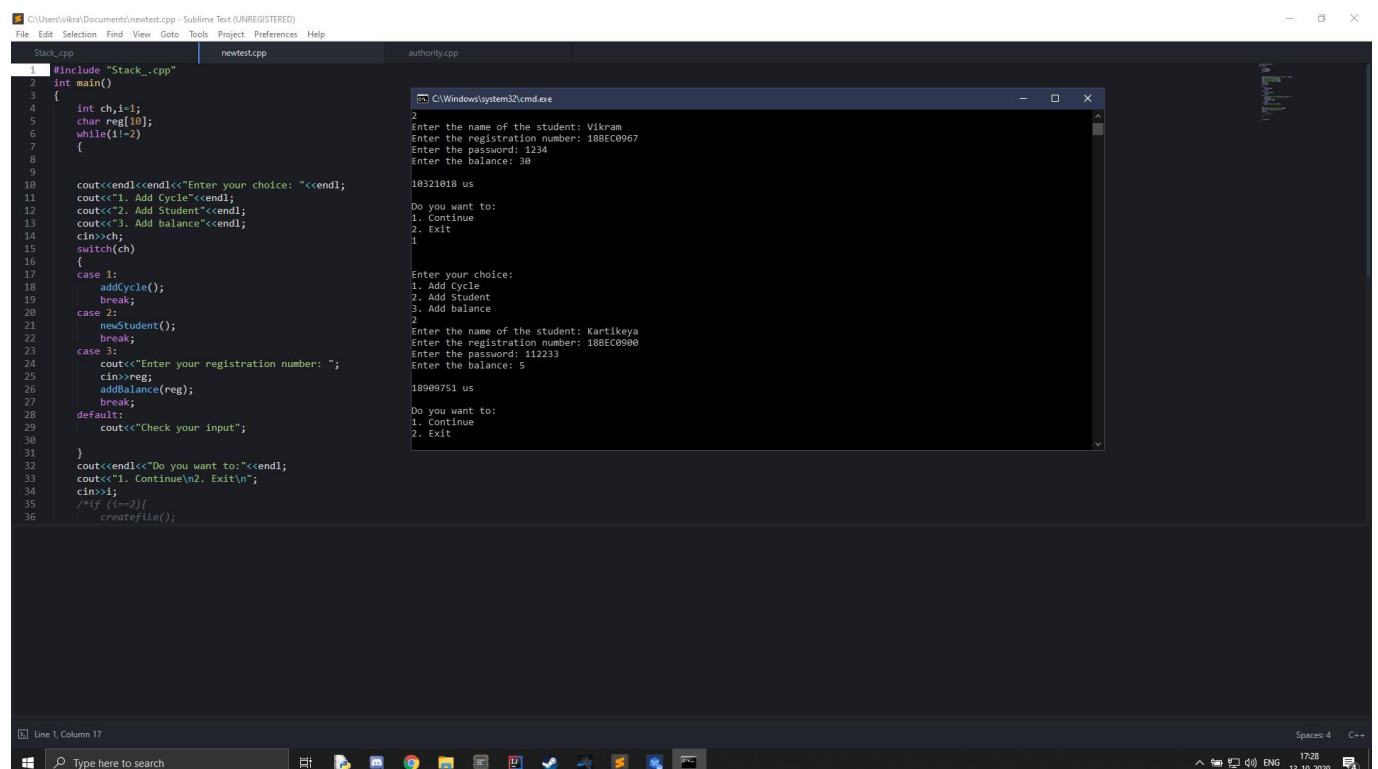
```
C:\Windows\system32\cmd.exe
Enter your choice:
1. Add Cycle
2. Add Student
3. Add balance
1
Enter the cycle registration number: rc123
Enter the cycle lock number: 123
Cycle added SUCCESSFULLY!!
2311798 us

Do you want to:
1. Continue
2. Exit
1

Enter your choice:
1. Add Cycle
2. Add Student
3. Add balance
1
Enter the cycle registration number: rc234
Enter the cycle lock number: 234
Cycle added SUCCESSFULLY!!
3487952 us

Do you want to:
1. Continue
2. Exit
```

### User



```
1 #include "Stack_.cpp"
2 int main()
3 {
4     int ch,i=1;
5     char reg[10];
6     while(i!=2)
7     {
8
9
10
11         cout<<endl<<"Enter your choice: "<<endl;
12         cout<<"1. Add Cycle"<<endl;
13         cout<<"2. Add Student"<<endl;
14         cout<<"3. Add balance"<<endl;
15         cin>>ch;
16         switch(ch)
17         {
18             case 1:
19                 addCycle();
20                 break;
21             case 2:
22                 newStudent();
23                 break;
24             case 3:
25                 cout<<"Enter your registration number: ";
26                 cin>>reg;
27                 addBalance(reg);
28                 break;
29             default:
30                 cout<<"Check your input";
31         }
32         cout<<endl<<"Do you want to:"<<endl;
33         cout<<"1. Continue\n2. Exit\n";
34         cin>>i;
35         /*if (i==2){
36             createfile();
37         }*/
38     }
39 }
```

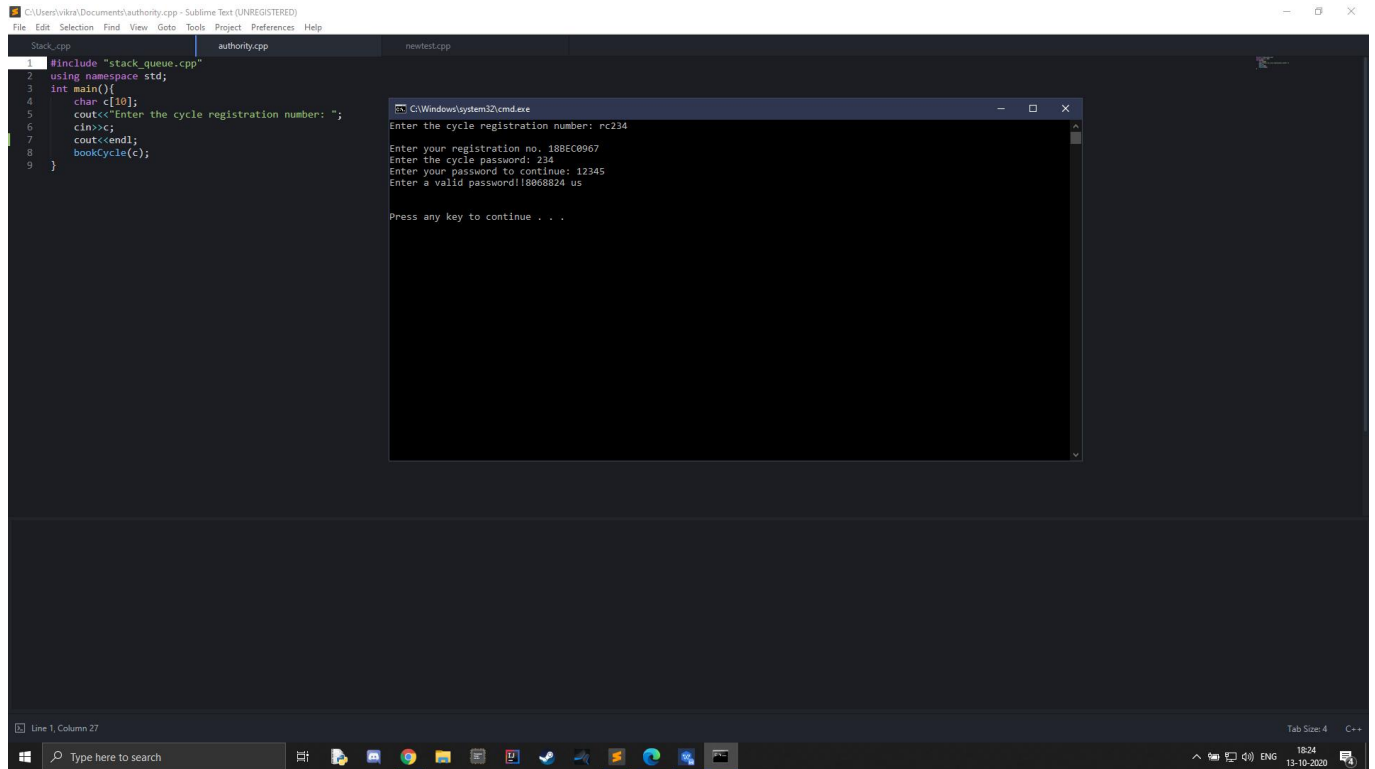
```
C:\Windows\system32\cmd.exe
2
Enter the name of the student: Vikram
Enter the registration number: 188EC0967
Enter the password: 1234
Enter the balance: 30
18321818 us

Do you want to:
1. Continue
2. Exit
1

Enter your choice:
1. Add Cycle
2. Add Student
3. Add balance
2
Enter the name of the student: Kartikeya
Enter the registration number: 188EC0960
Enter the password: 112233
Enter the balance: 5
18909751 us

Do you want to:
1. Continue
2. Exit
```

## Authority (with incorrect password)



The screenshot shows the Sublime Text editor with the file `authority.cpp` open. The code is as follows:

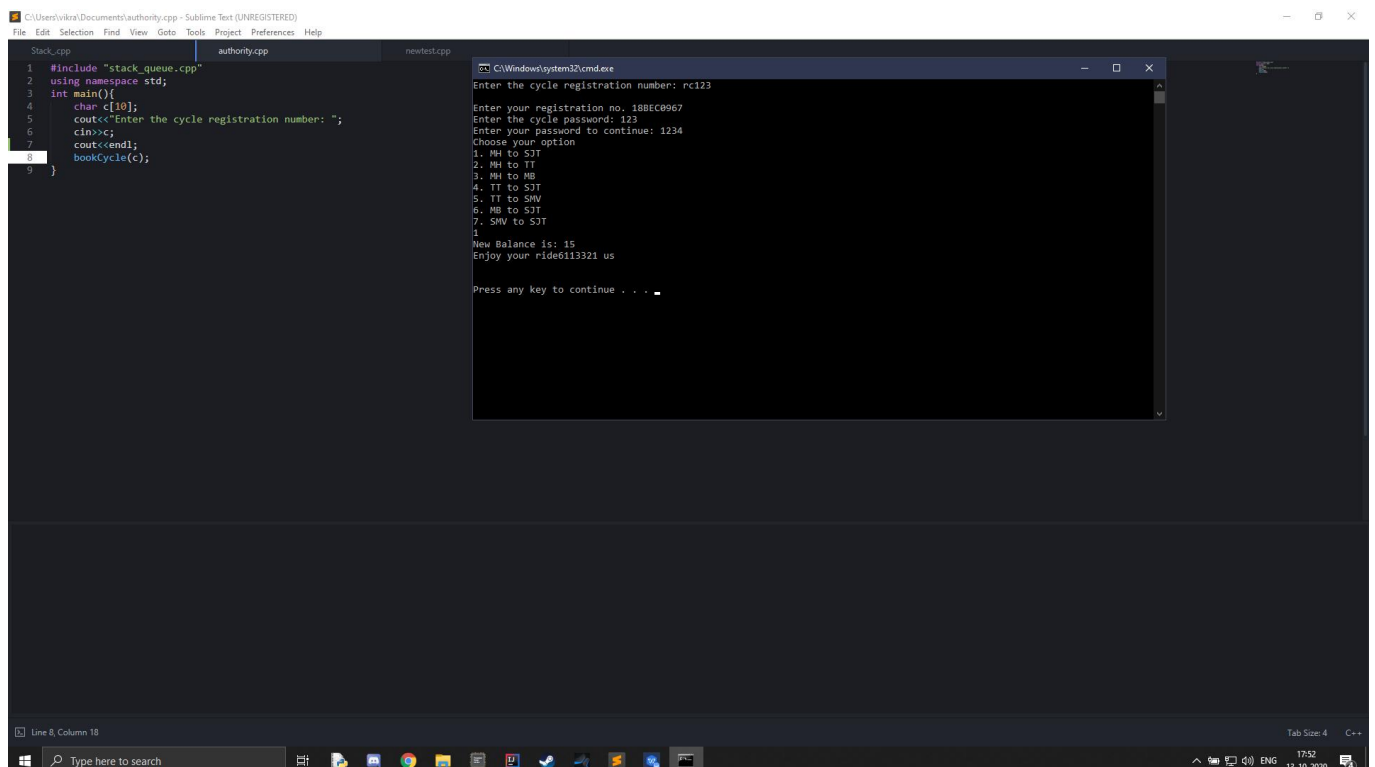
```
1 #include "stack_queue.cpp"
2 using namespace std;
3 int main()
4 {
5     char c[10];
6     cout<<"Enter the cycle registration number: ";
7     cin>>c;
8     cout<<endl;
9     bookCycle(c);
10 }
```

A Windows command prompt window is overlaid on the editor, showing the execution of the program. The user has entered the registration number `rc234` and the password `12345`. The program outputs the following:

```
Enter the cycle registration number: rc234
Enter your registration no. 188EC0967
Enter the cycle password: 234
Enter your password to continue: 12345
Enter a valid password!18866824 us

Press any key to continue . . .
```

## Authority (with correct password)



The screenshot shows the Sublime Text editor with the file `authority.cpp` open. The code is as follows:

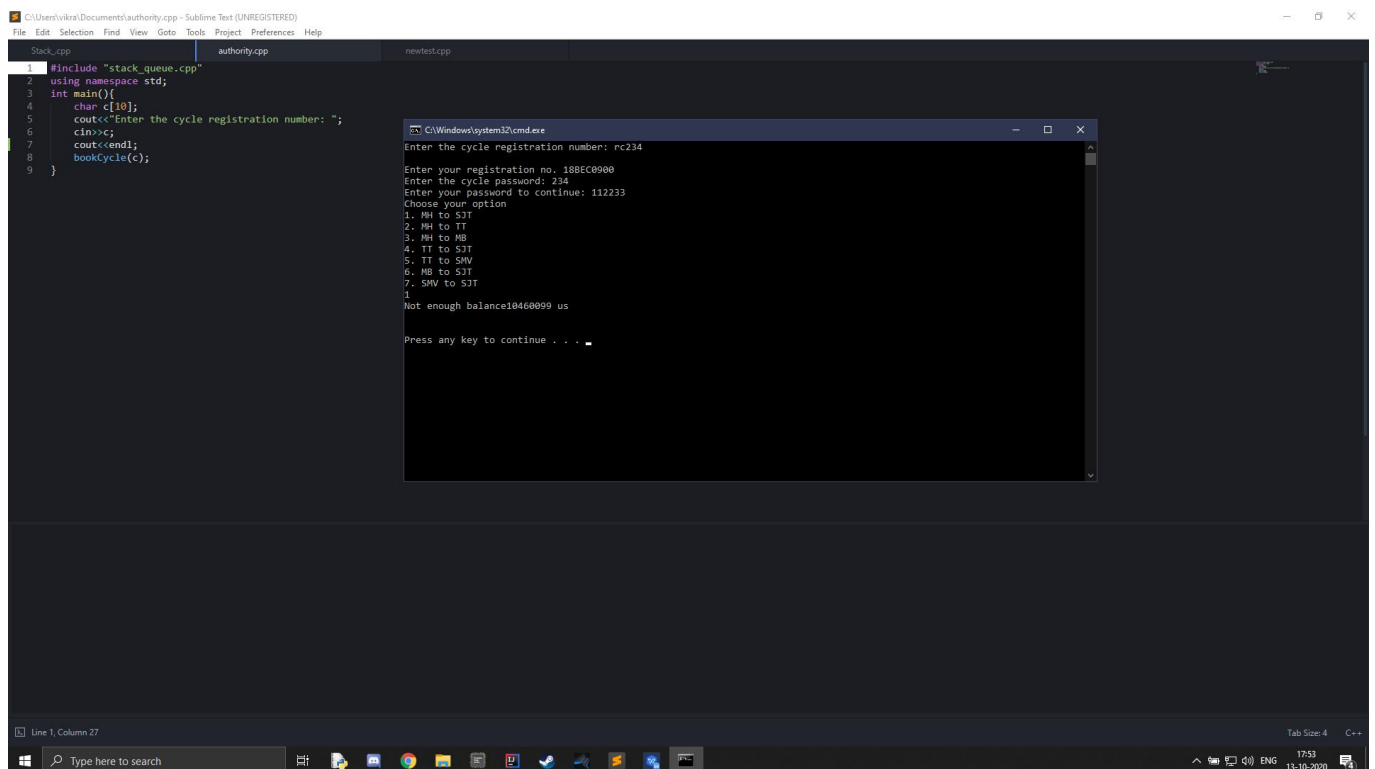
```
1 #include "stack_queue.cpp"
2 using namespace std;
3 int main()
4 {
5     char c[10];
6     cout<<"Enter the cycle registration number: ";
7     cin>>c;
8     cout<<endl;
9     bookCycle(c);
10 }
```

A Windows command prompt window is overlaid on the editor, showing the execution of the program. The user has entered the registration number `rc123` and the password `123`. The program outputs the following:

```
Enter the cycle registration number: rc123
Enter your registration no. 188EC0967
Enter the cycle password: 123
Enter your password to continue: 1234
Choose your option
1. MH to SJT
2. MH to TT
3. MH to MB
4. TT to SJT
5. TT to SMV
6. MB to SJT
7. SMV to SJT
1
New Balance is: 15
Enjoy your ride6113321 us

Press any key to continue . . .
```

## Authority for second user (where balance is less than trip rate)



The screenshot shows the Sublime Text editor with the file `authority.cpp` open. The code is as follows:

```
1 #include "stack_queue.cpp"
2 using namespace std;
3 int main()
4 {
5     char c[10];
6     cout<<"Enter the cycle registration number: ";
7     cin>>c;
8     cout<<endl;
9     bookCycle(c);
10 }
```

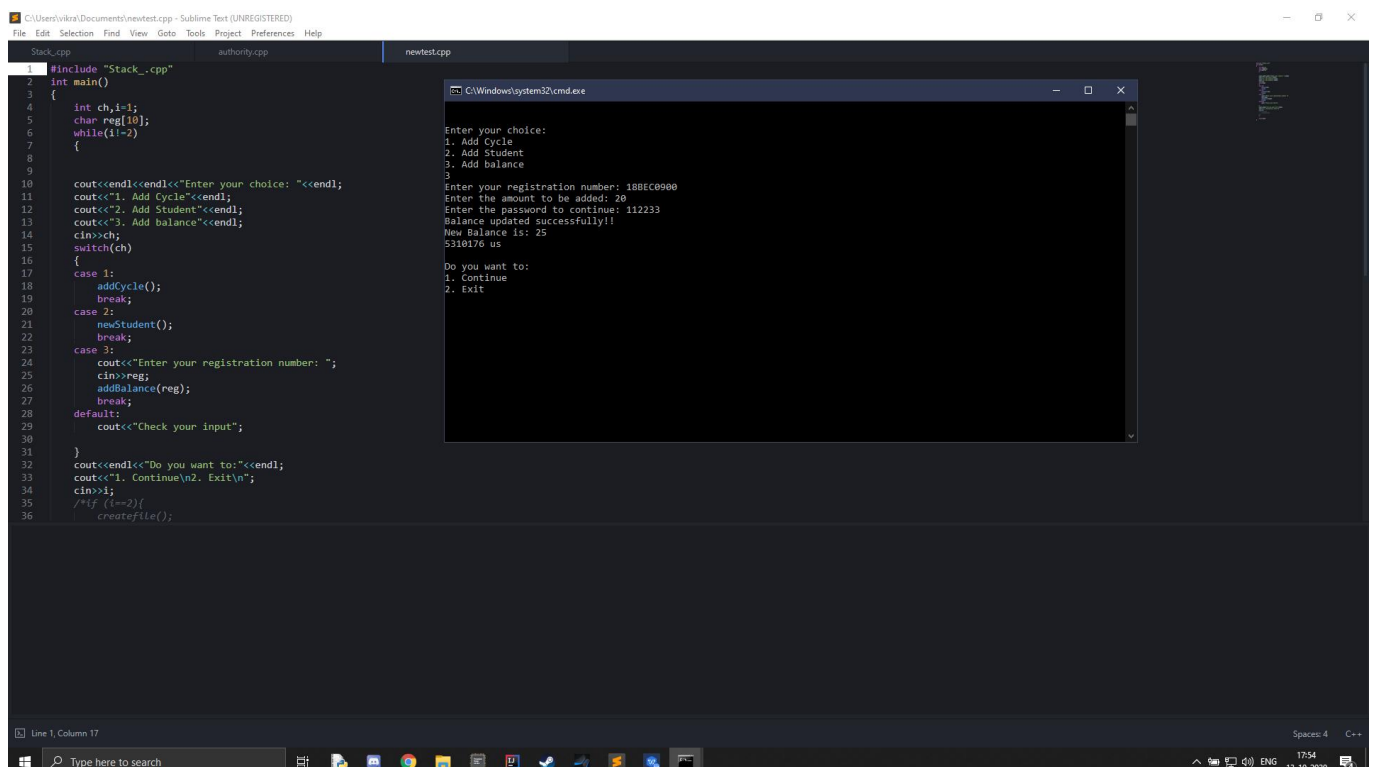
A Windows command prompt window is open, showing the execution of the program. The user has entered the cycle registration number `rc234`. The program prompts for a registration number, password, and then displays a menu of options. The user has chosen option 1, but the program outputs "Not enough balance" because the balance is less than the trip rate.

```
C:\Windows\system32\cmd.exe
Enter the cycle registration number: rc234

Enter your registration no, 18BEC0900
Enter the cycle password: 234
Enter your password to continue: 112233
Choose your option
1. MH to SJT
2. MH to TT
3. MH to MB
4. TT to SJT
5. TT to SHV
6. MB to SJT
7. SHV to SJT
1
Not enough balance10460099 us

Press any key to continue . . .
```

## Adding balance for second user



The screenshot shows the Sublime Text editor with the file `newtest.cpp` open. The code is as follows:

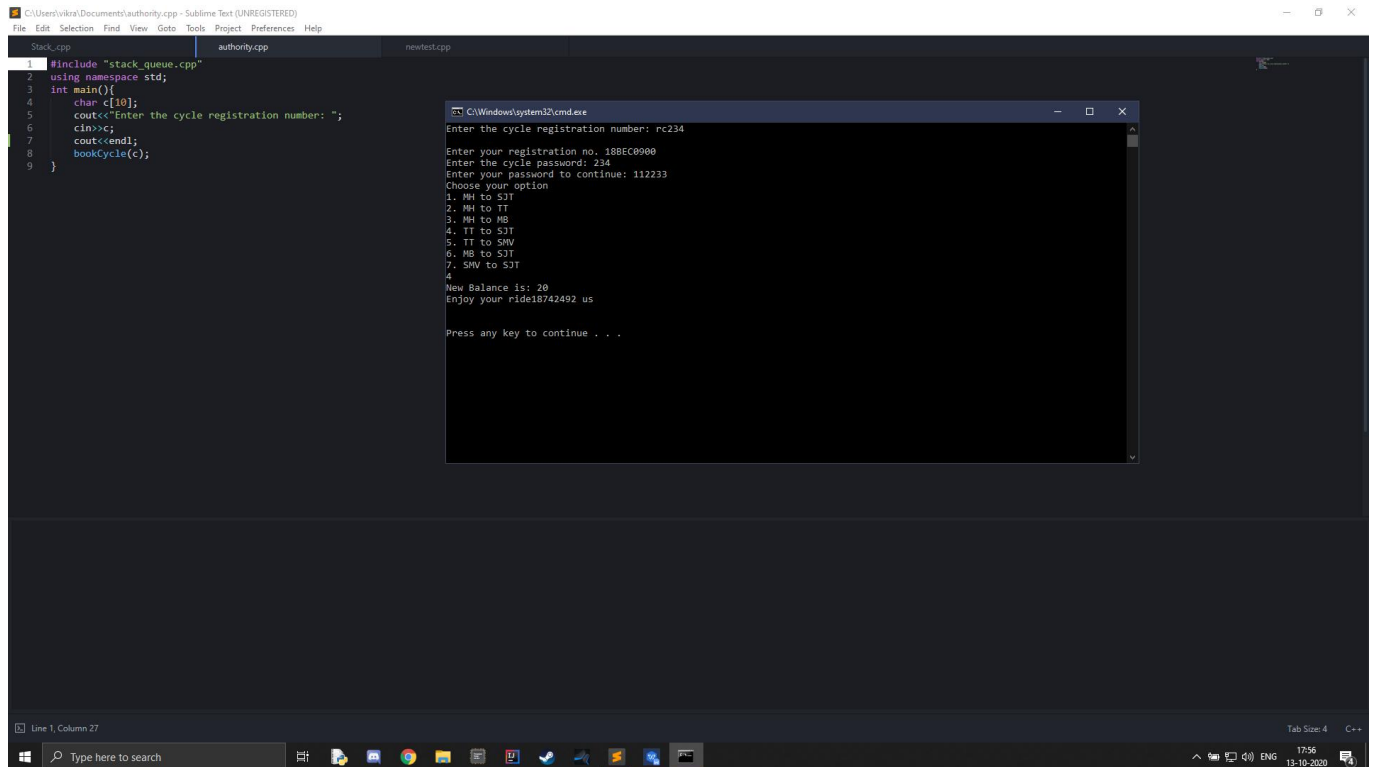
```
1 #include "Stack_.cpp"
2 int main()
3 {
4     int ch, i=1;
5     char reg[10];
6     while(i!=2)
7     {
8
9
10
11         cout<<endl<<endl<<"Enter your choice: "<<endl;
12         cout<<"1. Add Cycle"<<endl;
13         cout<<"2. Add Student"<<endl;
14         cout<<"3. Add Balance"<<endl;
15         cin>>ch;
16         switch(ch)
17         {
18             case 1:
19                 addCycle();
20                 break;
21             case 2:
22                 newStudent();
23                 break;
24             case 3:
25                 cout<<"Enter your registration number: ";
26                 cin>>reg;
27                 addBalance(reg);
28                 break;
29             default:
30                 cout<<"Check your input";
31         }
32         cout<<endl<<"Do you want to:"<<endl;
33         cout<<"1. Continue\n2. Exit\n";
34         cin>>i;
35         /*if (i==2){
36             createfile();
37         }*/
38     }
39 }
```

A Windows command prompt window is open, showing the execution of the program. The user has chosen option 3, entered the registration number `18BEC0900`, and entered the amount to be added `20`. The program outputs "Balance updated successfully!!" and "New Balance is: 25 5310170 us". The user is then prompted to choose between "1. Continue" and "2. Exit".

```
C:\Windows\system32\cmd.exe
Enter your choice:
1. Add Cycle
2. Add Student
3. Add Balance
3
Enter your registration number: 18BEC0900
Enter the amount to be added: 20
Enter the password to continue: 112233
Balance updated successfully!!
New Balance is: 25
5310170 us

Do you want to:
1. Continue
2. Exit
```

## Booking cycle for second user (after adding balance)



The screenshot shows a C++ program in Sublime Text and its execution in a Windows command prompt. The program is named `authority.cpp` and is located at `C:\Users\virel\Documents\authority.cpp`. The code is as follows:

```
1 #include "stack_queue.cpp"
2 using namespace std;
3 int main()
4 {
5     char c[10];
6     cout<<"Enter the cycle registration number: ";
7     cin>>c;
8     cout<<endl;
9     bookCycle(c);
10 }
```

The command prompt shows the execution of the program. The user enters the cycle registration number `rc234`. The program then prompts for the registration number, password, and password confirmation. The user enters `188EC0900` for the registration number, `234` for the password, and `112233` for the password confirmation. The program then prompts for a choice of option. The user enters `4`. The program then displays the new balance and the user's ride.

```
C:\Windows\system32\cmd.exe
Enter the cycle registration number: rc234
Enter your registration no. 188EC0900
Enter the cycle password: 234
Enter your password to continue: 112233
Choose your option
1. MH to SJT
2. MH to TT
3. MH to MB
4. TT to SJT
5. TT to SHW
6. MB to SJT
7. SHW to SJT
4
New Balance is: 20
Enjoy your ride!8742492 us
Press any key to continue . . .
```

## Results

Function	Stack Compile Time	Queue Compile Time
addCycle	2311798 us	3141788 us
newStudent	10321018 us	15762137 us
addBalance	5310176 us	8009088 us
bookCycle	18742492 us	14351961 us

## Conclusion

The compilation times are more or less the same (unit of time taken is in microseconds) as the time complexities of stack and queue both are  $O(1)$ .

However we see a slight difference in the compilation times as time complexity and space complexity also rely on other factors such as hardware, operating system, processors, etc.

Therefore we may get a different compilation time each time we run the program.

## References

1. Thomas H. Cormen , "Introduction to algorithms", The MIT press, McGraw-Hill Book Company.
2. Reinelt, Gerhard. The travelling salesman : computational solutions for TSP applications. Springer-Verlag, 1994.
3. <https://codelabs.developers.google.com/>
4. [www.geegsforgeeks.org](http://www.geegsforgeeks.org)
5. <https://www.androidauthority.com/>
6. <https://firebase.google.com/docs/>