# 物件導向程式設計

## Stack/Queue Template Class Case Study

Joseph Chuang-Chieh Lin
Dept. CSIE, Tamkang University

# Platform

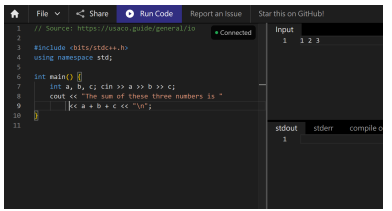
- Dev-C++

  Click here to download.

  **Note**: Please use this version otherwise you can't compile your programs/projects in Win10.

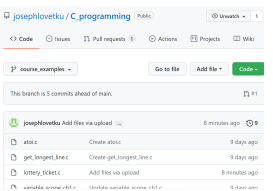

- OnlineGDB (https://www.onlinegdb.com/)

- Real-Time Collaborative Online IDE (https://ide.usaco.guide/)



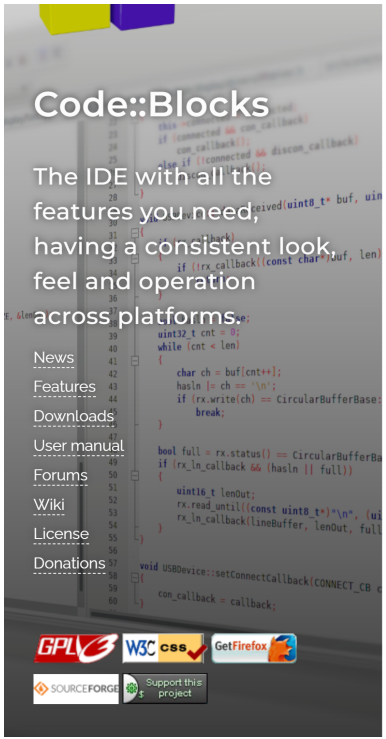

- Other resources:
  - MIT OpenCourseWare - Introduction to C++ [link].
  - Learning C++ Programming [Programiz].
  - GeeksforGeeks [link]

My GitHub page:
click the link here to visit.



# Platform/IDE

- https://www.codeblocks.org/




Code::Blocks

## Code::Blocks

### The free C/C++ and Fortran IDE.

Code::Blocks is a free C/C++ and Fortran IDE built to meet the most demanding needs of its users. It is designed to be very extensible and fully configurable.

Built around a plugin framework, Code::Blocks can be extended with plugins. Any kind of functionality can be added by installing/coding a plugin. For instance, event compiling and debugging functionality is provided by plugins!

If you 're new here, you can read the **user manual** or visit the **Wiki** for documentation. And don't forget to visit and join our **forums** to find help or general discussion about Code::Blocks.

We hope you enjoy using Code::Blocks!

*The Code::Blocks Team*

## Latest news

### Migration successful

We are very happy to announce that the process of migrating to the new infrastructure has completed successfully!

Read more

# Stack

# Stack Implementation (First-In-Last-Out)

https://www.geeksforgeeks.org/implementing-stack-using-class-templates-in-cpp/

```
template <class T, int SIZE> class Stack {
public:
    Stack();
    void push(T k);
    T pop();
    T topElement();
    bool isFull();
    bool isEmpty();

private:
    int top;
    T st[SIZE];
};
```

https://onlinegdb.com/TSHMlcLE9

# Constructor & `push()`

```
template <class T, int SIZE> Stack<T, SIZE>::Stack() { top = -1; }
```

```
template <class T, int SIZE> void Stack<T, SIZE>::push(T k) {
    if (isFull()) {
        cout << "Stack is full\n";
    } else {
        cout << "Inserted element " << k << endl;
        top = top + 1;
        st[top] = k;
    }
}
```

# IsEmpty() & IsFull()

```
template <class T, int SIZE> bool Stack<T,SIZE>::isEmpty() {
    if (top == -1)
        return 1;
    else
        return 0;
}
```

```
template <class T, int SIZE> bool Stack<T, SIZE>::isFull() {
    if (top == (SIZE - 1))
        return 1;
    else
        // As top can not exceeds th size
        return 0;
}
```

# topElement()

```
template <class T, int SIZE> T Stack<T, SIZE>::topElement() {
    T top_element = st[top];
    return top_element;
}
```

# The driver `main()`

```
int main() {
    int size = 10;
    cout << "Enter the size of the stack: ";
    Stack<int, size> integer_stack;
    Stack<string, size> string_stack;

    integer_stack.push(5);
    integer_stack.push(25);
    integer_stack.push(125);

    string_stack.push("COVID-19");
    string_stack.push("PANDEMIC");
    string_stack.push("FLU");

    cout << integer_stack.pop() << " is popped from the integer stack." << endl;
    cout << string_stack.pop() << " is popped from the string stack " << endl;
    cout << "Top element is " << integer_stack.topElement() << endl;
    cout << "Top element is " << string_stack.topElement() << endl;
    return 0;
}
```

```
Inserted element PANDEMIC
Inserted element FLU
125 is popped from the integer stack.
FLU is popped from the string stack
Top element is 25
Top element is PANDEMIC
```

# Queue

# Queue Implementation (First-In-First-Out)

https://slaystudy.com/c-program-to-implement-queue-using-templates/

```
template <class T>
class node {
public:
    T v;
    node<T> *next;
    node(T x) { // constructor
        v = x;
        next = nullptr;
    }
};
```

```
template <class T>
class queue {
    node<T> *start;
    node<T> *end;

public:
    queue();
    bool empty();
    void push(T v); // enqueue
    T front(); // peek
    void pop(); // dequeue
};
```

# Constructor & `empty()`

```
template <class T> queue<T>::queue() {
    start = nullptr;
    end = nullptr;
}
```

```
template <class T> bool queue<T>::empty() {
    return start == nullptr;
}
```

# push()

```
template <class T> void queue<T>::push(T v) {
    node<T> *temp = new node<T>(v);
    if (empty())   {
        start = end = temp;
    } else {
        end->next = temp;
        end = temp;
    }
}
```

# front() & pop()

```
template <class T> T queue<T>::front() {
    if (empty())
        return nullptr;
    else
    return start->v;
}
```

```
template <class T>
void queue<T>::pop() {
    if (empty()) {
            cout << "Queue is Empty" << endl;
    } else if (start == end) {
        delete start;
        start = nullptr;
        end = nullptr;
    } else {
        node<T> *temp = start;
        start = start->next;
        delete temp;
    }
}
```

# The driver `main()`

```
int main() {
    queue<string> q;
    q.push("TKU");
    q.push("CSIE");

    cout << "Queue Front: " << q.front() << endl;

    q.push("Definitely");
    q.push("I");
    q.push("am");
    q.push("the");
    q.push("best");

    while (!q.empty()) {
        cout << q.front() << ' ';
        q.pop();
    }
    return 0;
}
```

```
Queue Front: TKU
TKU CSIE Definitely I am the best
```