NSUPS Bootcamp Week 4

Introduction to Data Structure

First, some I/O tips

1. How do we read input from a file?

   The following reads input from a file named “input.txt”

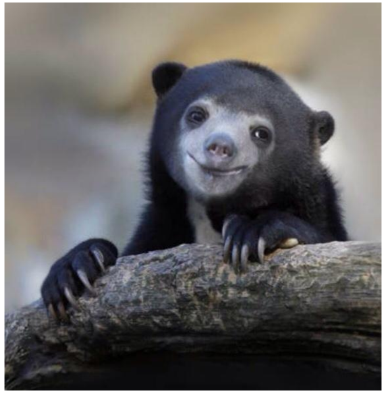
   ```
   freopen(“input.txt”, “r”, stdin);
   ```

2. How do we write output to a file?

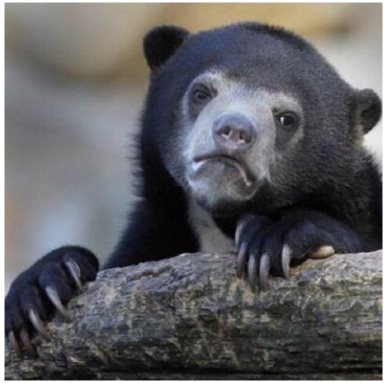
   The following writes output to a file named “output.txt”

   ```
   freopen(“output.txt”, “w”, stdout);
   ```

No more Number Theory!



Just Kidding! There’s plenty more left to learn!



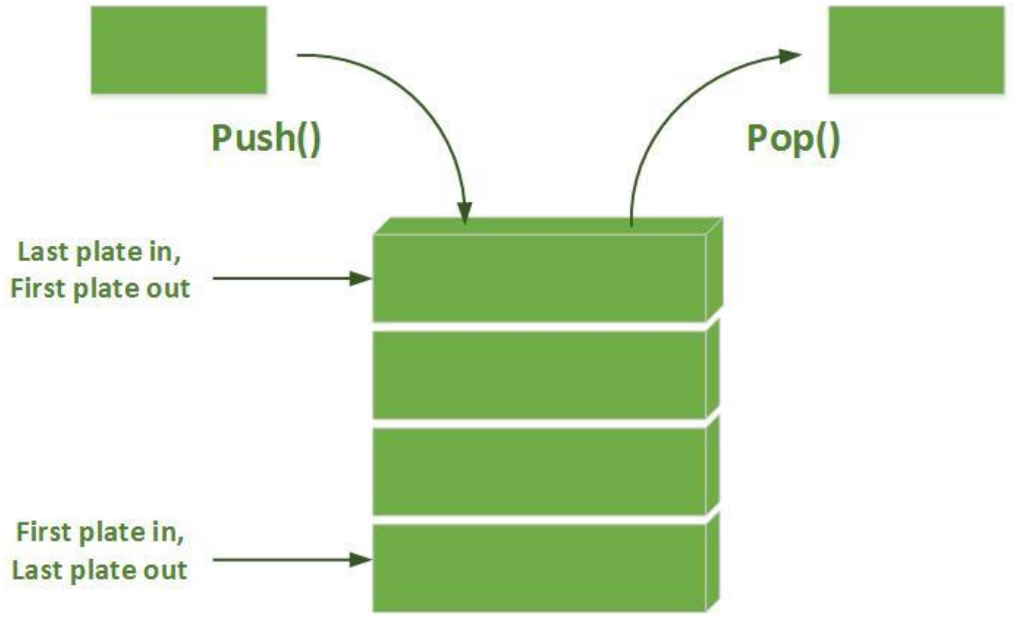
But this is enough for you to get started.
Now we start learning some Data Structure!

C++ STL

1. vector
2. set
3. map
4. string operations

Stack

Stack is a data structure where elements are inserted in and extracted out of only one end of the container. How do we code it?



STL : Stack

1. C++ has a built-in class that does the work of Stack.
2. You can push elements into a stack.
3. You can pop elements from the top of the stack.
4. You can get the last inserted element of the stack.

STL : Stack

```
#include <iostream>
#include <stack>

using namespace std;

void showstack(stack <int> gq)
{
    stack <int> g = gq;
    while (!g.empty())
    {
        cout << '\t' << g.top();
        g.pop();
    }
    cout << '\n';
}

//-------------- Output ---------------
The stack gquiz is : 1 5 20 30 10
gquiz.size() : 5
gquiz.top() : 1
gquiz.pop() : 5 20 30 10
```

```
int main ()
{
    stack <int> gquiz;
    gquiz.push(10);
    gquiz.push(30);
    gquiz.push(20);
    gquiz.push(5);
    gquiz.push(1);

    cout << "The stack gquiz is : ";
    showstack(gquiz);

    cout << "\ngquiz.size() : " << gquiz.size();
    cout << "\ngquiz.top() : " << gquiz.top();


    cout << "\ngquiz.pop() : ";
    gquiz.pop();
    showstack(gquiz);

    return 0;
}
```

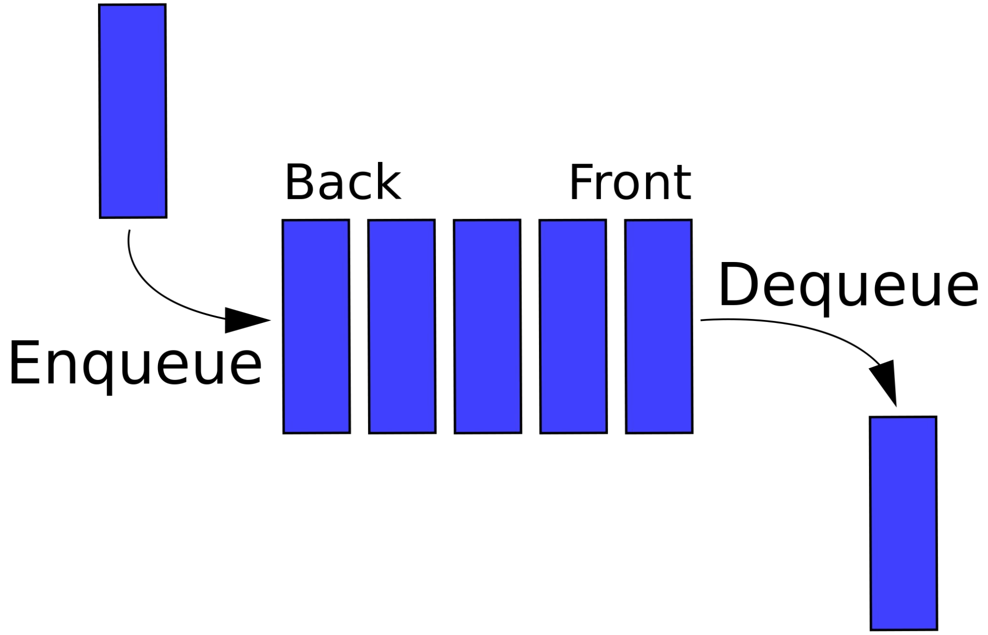
Some problems you can solve using Stack

1. Minimum value in the stack
2. Parenthesis Balance
    a. When the are of same kind (())()
    b. Multiple kind: (){[]}({})
3. Nearest Smaller value in array

A problem from this ICPC!

Queue

Queue is a data structure where elements are inserted in the back and extracted out from the front. How do we code a queue?



STL : Queue

1. C++ has a built-in class that does the work of Queue.
2. You can push elements in the back of the Queue.
3. You can pop elements from the front of the Queue.
4. You can get the oldest element in the queue (the element in front).

STL : Queue

```
#include <iostream>
#include <queue>

using namespace std;

void showq(queue <int> gq)
{
    queue <int> g = gq;
    while (!g.empty())
    {
        cout << '\t' << g.front();
        g.pop();
    }
    cout << '\n';
}
//--------------     Output     -------------------

The queue gquiz is :   10    20

gquiz.size()                  :                  2

gquiz.front()                 :                 10

gquiz.back()                 :                  20

gquiz.pop() :    20
```

```
int main()
{
    queue <int> gquiz;
    gquiz.push(10);
    gquiz.push(20);

    cout << "The queue gquiz is : ";
    showq(gquiz);

    cout << "\ngquiz.size() : " << gquiz.size();
    cout << "\ngquiz.front() : " << gquiz.front();
    cout << "\ngquiz.back() : " << gquiz.back();

    cout << "\ngquiz.pop() : ";
    gquiz.pop();
    showq(gquiz);

    return 0;
}
```

Dequeue

1. Similar to queue. But now we can perform both insert and delete operations in both back and front side.
2. *front()* returns front element like queue
3. *pop_front()* erase the front element
4. *back()* returns the back element
5. *pop_back()* erase the back element
6. How to solve 1093 - Ghajini

Resources

1. STL Tutorials
   http://www.geeksforgeeks.org/the-c-standard-template-library-stl/
2. Sorting Data
   http://en.cppreference.com/w/cpp/algorithm/sort
3. freopen tutorial
   http://www.cplusplus.com/reference/cstdio/freopen/