

Stack (3A)

Copyright (c) 2015 Young W. Lim.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using OpenOffice and Octave.

Separate Compile

```
int sum(int, int);
```

```
int sum(int x, int y) {  
    return (x + y);  
}
```

```
#include <stdio.h>  
#include "t.h"  
  
int main (void) {  
    int a = 10;  
    int b = 20;  
  
    int c;  
  
    c = sum(10, 20);  
  
    printf("a=%d b=%d c=%d \n", a, b, c);  
  
    return 0;  
}
```

Stack API

```
typedef struct aaa node;
typedef struct aaa {
    int Data;
    node *Next;
} node;

typedef node* Nptr;

class stackClass {
public:
    ~stackClass (); // destructor function
    stackClass (); // default constructor
    stackClass (const stackClass& S); // constructor

    void Push (int Item);
    int Pop ();
    int IsEmpty ();
    int IsFull ();

private:
    Nptr Top;
};
```

Stack Implementation

```
#include "StackP.hpp"
#include <iostream>
#include <stdio.h>

using namespace std;

stackClass::~~stackClass() {
    while ( !IsEmpty() ) {
        Pop();
    }
}

stackClass::stackClass() {
    Top = NULL;
}

stackClass::stackClass(const stackClass& S) {

}

void stackClass::Push(int Item) {
    Nptr NewTop = new node;

    NewTop->Data = Item;
    NewTop->Next = Top;
    Top = NewTop;
}

int stackClass::Pop() {
    if ( IsEmpty() ) {
        printf("Deletion on Empty Stack \n");
        cout << "Deletion on Empty Stack \n";
        return -1;
    } else {
        Nptr Temp = Top;
        int Item = Temp->Data;
        Top = Top-> Next;

        delete Temp;
        return Item;
    }
}

int stackClass::IsEmpty() {
    return (Top == NULL);
}

int stackClass::IsFull() {
    return 0;
}
```

Function Calls (1)

```
#include <stdio.h>

void func1(int n) {
    printf("func1: n: %d \n", n);
    n += 10;
    printf("func1: n: %d \n", n);
    printf("func1: &n: %p \n", &n);
}

void func2(int* n) {
    printf("func2: *n: %d \n", *n);
    *n += 10;
    printf("func2: *n: %d \n", *n);
    printf("func2: &n: %p \n", &n);
    printf("func2: n: %p \n", n);
}

int main(void) {
    int a = 10;

    printf("----call by value -----\n" );
    printf("main: &a: %p \n", &a);
    printf("main: a: %d \n", a);
    func1( a );
    printf("main: a: %d \n", a);
    printf("\n");

    printf("----call by reference -----\n" );
    printf("main: &a: %p \n", &a);
    printf("main: a: %d \n", a);
    func2( &a );
    printf("main: a: %d \n", a);
    printf("\n");
}
```

Function Calls (2)

```
#include <stdio.h>

void func3(int& n) {
    printf("&n: %p \n", &n);
    printf("n: %d \n", n);
}

int main(void) {
    int a = 10;
    int& b = a;

    int * p = &a;

    printf("&a: %p \n", &a);
    printf("a: %d \n", a);

    printf("&b: %p \n", &b);
    printf("b: %d \n", b);

    func3( a );

    printf("&p: %p \n", &p);
    printf("p: %p \n", p);
    printf("*p: %d \n", *p);
}
```

Modulo

```
#include <stdio.h>

int main(void) {
    int i, j, k;

    for (i=-10; i<10; ++i) {
        j = i % 4;
        k = ((i % 4)+4) % 4;
        printf("%d %% 4 = %d  %d \n", i, j, k );
    }
}
```


Dynamic Memory Allocation

```
#include <stdio.h>

int main(void) {
    int i, j, k;

    for (i=-10; i<10; ++i) {
        j = i % 4;
        k = ((i % 4)+4) % 4;
        printf("%d %% 4 = %d %d \n", i, j, k );
    }
}
```

Array of Structure

```
#include <stdio.h>

using namespace std;

struct aaa {
    int i;
    short s;
    char c;
};

typedef struct aaa M;

void pr(M x) {
    printf("member i= %d \n", x.i);
    printf("member s= %d \n", x.s);
    printf("member c= %c \n", x.c);
}

void pr2(M& x) {
    printf("member i= %d \n", x.i);
    printf("member s= %d \n", x.s);
    printf("member c= %c \n", x.c);
}

void pr3(M* x) {
    printf("member i= %d \n", x->i);
    printf("member s= %d \n", x->s);
    printf("member c= %c \n", x->c);
}
```

```
int main(void) {

    M arr[5];

    arr[0].i = 10;
    arr[0].s = 1;
    arr[0].c = 'a';

    arr[1].i = 20;
    arr[1].s = 2;
    arr[1].c = 'b';

    pr(arr[0]);

    pr(arr[1]);

    pr2(arr[0]);
    pr2(arr[1]);

    pr3(&arr[0]);
    pr3(&arr[1]);
}
```

Stack Implementation

Stack Implementation

Stack Implementation

Stack Implementation

Stack Implementation

Stack Implementation

Stack Implementation

References

- [1] <http://en.wikipedia.org/>
- [2]