STACK STRUCTURE

Implementing a stack as a C++ structure

Delroy A. Brinkerhoff

STACK HEADER FILE

Stack Structure

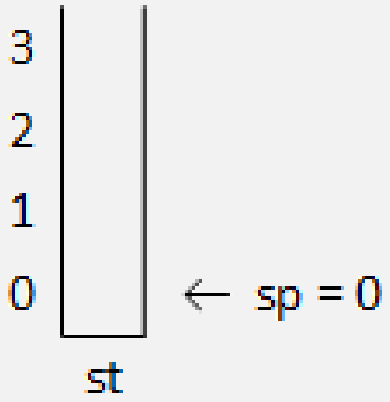
Stack Function Prototypes

```
//#define SIZE 100
//enum { SIZE = 100; }
const int SIZE = 100;

struct stack
{
        char   st[SIZE];
        int    sp;
};

stack  make_stack();
void   init_stack(stack* s);
void   push(stack* s, char data);
char   pop(stack* s);
int    size(stack* s);
char   peek(stack* s);
```

INSTANTIATING AND INITIALIZING




```
stack make_stack()
{
    stack  temp;
    temp.sp = 0;
    return temp;
}

void init_stack(stack* s)
{
   s->sp = 0;
}
```

REQUIRED OPERATIONS

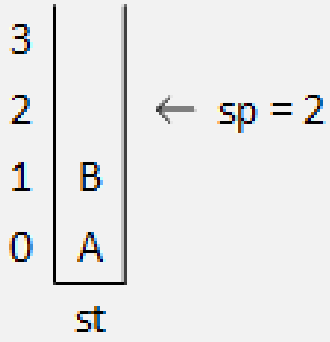
```
void push(stack* s, char data)
{
    if (s->sp < SIZE)
        s->st[s->sp++] = data;
    else
        throw "Stack Overflow";
}
```

```
char pop(stack* s)
{
    if (s->sp > 0)
        return s->st[--(s->sp)];
    else
        throw "Stack Underflow";
}
```

OPTIONAL OPERATIONS

```
int size(stack* s)
{
    return s->sp;
}
```

```
char peek(stack* s)
{
    return s->st[s->sp - 1];
}
```



MAKING AND USING A STACK

A simple client

```
#include <iostream>
#include "stack.h"
using namespace std;

int main()
{
        stack s;
        init_stack(&s);

        push(&s, 'x');
        push(&s, 'y');
        push(&s, 'z');

        char c = peek(&s);
        cout << c << endl;

        c = pop(&s);
        cout << c << endl;

        while (size(&s) > 0)
                cout << pop(&s) << endl;

        return 0;
}
```