// comsc 200

// main.cpp

// as10

// boli zhang

// Created by Jeff on 10/24/16.

// Copyright © 2016 Jeff zhang. All rights reserved.

//

#include <iostream>

#include <string>

#include <sstream>

#include <cstring>

using namespace std;

template<class T>

class Stack{

private:

T \*stackArray;

int stackCapacity;

int top;

public:

class Excep\_over{ };

class Excep\_Under{ };

Stack(int size){

// dynimic assement

stackArray = new T[size];

stackCapacity =size;

top = -1;

}

~Stack(){

delete[]stackArray;

}

// destructor();

int lengthoftop(){

return top;

}

void push(T item){

if(isFull())

// cout<<" .ERR.The stack is full."<<endl;

throw Excep\_over();

else

{

top++;

stackArray[top] = item;

}

}

void pop(T &item){

if(isEmpty()){

throw Excep\_Under();

}

else{

item = stackArray[top];

top--;

}

}

void popnum(T &item,string num){

if(isEmpty()){

throw Excep\_Under();

}

else{

item = stackArray[top];

top--;

}

}

bool isFull(){

if(top >= stackCapacity - 1) return true;

else

return false;

}

bool isEmpty(){

if(top== -1) return true;

return false;

}

void display(){

if(isEmpty())

cout << ".ERR.The stack is empty."<<endl;

else{

cout << ".STK. ";

for(int i =0 ; i <=top;i++){

cout << stackArray[i]<<",";

}

}

}

void displaynum(int n){

if(isEmpty())

cout << ".ERR.The stack is empty."<<endl;

else{

cout << "Stack [" << n << "] contains: ";

cout << stackArray[n] << endl;

}

}

};

void strLower(string &str);

void menu() {

cout<<"--- Assignment\_10 Stack Test Menu --- \n"

<< "N - to bulk create New stack \n"

<< "D - to Display \n"

<< "E - is Empty? \n"

<< "F - is Full? \n"

<< "P - to Push \n"

<< "R - to Pop (Remove)\n"

<< "G - to Get an entry (by subscription)\n"

<< "S - to Set an entry (by subscription)\n"

<< "L - Length of Stack?\n"

<< "C - to Clear\n"

<< "Q - to Q this proragm \n";

}

int main(int argc, const char \* argv[]) {

menu();

string catchVar; // To hold values popped off the stack

Stack<string> stk(6);

stk.push("This");

stk.push("is");

stk.push("an");

stk.push("interesting");

stk.push("list !");

while(true){

string option; // user command

cout << "\n--- Enter a command -> ";

getline(cin, option);

strLower(option);

if(option == "n") {

string input, token;

cout << "Enter a line of comma (,) delmited data set: ";

getline (cin, input); // user input -> string

stringstream ss(input); // string -> stream

while ( getline(ss, token, ',') ) { // stream -> string token

stringstream sst(token); // string token -> stream token

string ff;

sst >> ff; // stream token -> double token

try{

stk.push(ff);

}catch(Stack<string>::Excep\_over){

cout<<" .ERR.The stack is full."<<endl;

}

}

}

else if(option == "d") {

stk.display();

}

else if(option == "r"){

try{

stk.pop(catchVar);

}catch(Stack<int>::Excep\_Under){

cout << ".ERR.The stack is empty."<<endl;

} stk.display();

}

else if( option == "p" ){

string astring;

cout <<"Enter a string token: \n";

getline(cin, astring);

try{

stk.push(astring);

}catch(Stack<string>::Excep\_over){

cout<<" .ERR.The stack is full."<<endl;

}

stk.display();

}

else if( option == "g"){

int num;

cout << "item position: \n" ;

cin >> num;

stk.displaynum(num);

}

else if(option == "s") {

string num ;

cout << "item position: ";

getline(cin, num);

string newvalue ;

cout <<"what is new value: \n";

getline(cin, newvalue);

stk.popnum(catchVar,num);

stk.push(newvalue);

}

else if(option == "c"){

int lengtharray=stk.lengthoftop();

for(int i=0; i<=lengtharray; i++) {

stk.pop(catchVar);

}

}

else if(option == "l"){

cout << "Number of Entries in Stack: \n";

cout << stk.lengthoftop();

}

else if(option == "e"){

if( !stk.isEmpty()) cout << "List Is not Empty \n";

else

cout << "list is empty \n";

}

else if(option == "f"){

if(!stk.isFull()) cout << "list is not Full \n";

else

cout << "list is full \n";

}

else if(option == "q"){

break;

}

} // end main menu while

return 0;

}

void strLower(string &str) {

locale loc;

string result;

for (int i=0; i<str.length(); ++i) {

result.push\_back(tolower(str[i],loc));

}

str = result;

}

