Lab 3

# Ptr.cpp

#include "pch.h"

#include <iostream>

using namespace std;

struct RecType

{

int age;

float weight;

char gender;

};

int main()

{

int var = 5;

int \*intPtr = NULL;

//intPtr = &var;

cout << \*intPtr \* 10 << endl;

RecType \*recPtr = new RecType;

recPtr->age = 25;

recPtr->weight = 190;

recPtr->gender = 'M';

(\*recPtr).age += 5;

cout << (\*recPtr).age << " " << (\*recPtr).weight << " " << (\*recPtr).gender << endl;

delete recPtr;

recPtr = NULL;

char \*strPtr = new char[50];

//strcpy was not used because it is deemed unsafe and produced compile errors

//strcpy\_s is the function to use here according to c++ documentation on the microsoft website

//note: i am using visual studio as my ide

strcpy\_s(strPtr, 18,"Operating Systems");

cout << strPtr << endl;

int lower = 0;

for (int i = 0; i < strlen(strPtr); i++)

{

if (islower(\*(strPtr + i)))

lower++;

}

cout << lower << endl;

strPtr += 10;

cout << strPtr << endl;

strPtr -= 10;

delete strPtr;

strPtr = NULL;

return 0;

}

# funcPtr.cpp

#include <iostream>

using namespace std;

bool areIdenticalValues(double \*ptr1,double \*ptr2);

bool areIdenticalAddresses(double \*ptr1, double \*ptr2);

int main()

{

double one = 12.8;

double two = 10.9;

double \*ptr1 = &one;

double \*ptr2 = &two;

//both return false

cout << areIdenticalValues(ptr1, ptr2) << endl;

cout << areIdenticalAddresses(ptr1, ptr2) << endl;

//identical values

\*ptr1 = 10.9;

cout << areIdenticalValues(ptr1, ptr2) << endl;

cout << areIdenticalAddresses(ptr1, ptr2) << endl;

//identical addresses, which leads to identical values

//this causes a memory leak because now 12.8 is inaccessible

ptr1 = ptr2;

cout << areIdenticalValues(ptr1, ptr2) << endl;

cout << areIdenticalAddresses(ptr1, ptr2) << endl;

return 0;

}

bool areIdenticalValues(double \*ptr1 ,double \*ptr2)

{

if (\*ptr1 == \*ptr2)

return true;

return false;

}

bool areIdenticalAddresses(double \* ptr1, double \* ptr2)

{

if (ptr1 == ptr2)

return true;

return false;

}

# stl.cpp

#include <string>

#include <vector>

#include <algorithm>

#include <iostream>

#include <stack>

using namespace std;

void initialize(vector<int> &v);

void reverse(vector<int> &v);

void print(vector<int> v);

int main()

{

stack<string> s1, s2;

s1.push("Hi there");

s2.push("Bye, there");

if (s1 == s2)

cout << "s1 and s2 are equal." << endl;

else if(s1<s2)

cout << "s1 is less than s2." << endl;

else if (s1 > s2)

cout << "s1 is greater than s2." << endl;

vector<int> v;

initialize(v);

print(v);

//the reverse function could have also been used here

//the lab instructions didn't specify to use the reverse function

//code below if reverse function was used

//reverse(v.begin(), v.end());

reverse(v);

print(v);

sort(v.begin(), v.end());

print(v);

return 0;

}

void initialize(vector<int> &v)

{

int temp;

for (int i = 0; i < 5; i++)

{

cout << "Please enter the number for position " << i << ":" << endl;

cin >> temp;

v.push\_back(temp);

}

}

void print(vector<int> v)

{

cout << "The vector will be printed below." << endl;

for (int i = 0; i < v.size() ; i++)

cout << "The value at position " << i << " is " << v[i] << endl;

}

void reverse(vector<int> &v)

{

vector<int> temp = v;

for (int i = 0; i < v.size(); i++)

{

v[i] = temp[5-i-1];

}

cout << "The vector has been reversed." << endl;

}