Task 1:

#include <iostream>

#include<iomanip>

using namespace std;

class Vector

{

public:

Vector(int s = 0) { // makes Size = s, //allocates s space, // makes all entries 0

size = s;

entries = new int[size];

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

entries[i] = 0;

}

}

Vector(const Vector & rhs) { // copy constructor // makes self a deep copy of rhs

size = rhs.size;

entries = new int[rhs.size];

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

entries[i] = rhs.entries[i];

}

}

Vector operator = (const Vector & rhs) {// makes self a deep copy of rhs

size = rhs.size;

entries = new int[rhs.size];

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

entries[i] = rhs.entries[i];

}

return \*this;

}

~Vector() { // default destructor

delete[] entries;

}

int & operator[](int index) { // if 0 <=pos<size // returns entries[index]

if (0 <= index && index < size) {

return entries[index];

}

else {

return entries[0];

}

}

int sz() { // returns the # of entries.

return size;

}

private:

int size; //store the # of elements used

int \*entries;

};

ostream & operator<<(ostream & out, Vector & rhs) {

//Vector walker = rhs;

for (int i = 0; i < rhs.sz(); i++) {

out << rhs[i] << " ";

}

return out;

}

int main() {

Vector test(10);

Vector \*vecPtr;

Vector exam(5);

for (int i = 0; i < test.sz(); i++) {

test[i] = i;

}

cout << "test initally is : ";

cout << test << endl;

cout << "empty Vector of length 5 is : ";

cout << exam << endl;

exam = test;

cout << "after exam = test, exam is : ";

cout << exam << endl;

{

Vector sqrs(10);

cout << "The squares are : ";

for (int i = 0; i < sqrs.sz(); i++)

sqrs[i] = i\*i;

vecPtr = &sqrs;

cout << sqrs << endl;

}

cout << "trying to print an empty vector: ";

cout << \*vecPtr << endl;

for (int i = 0; i < test.sz(); i++)

test[i] = 10 - i;

cout << "testing deep copy :\n";

cout << "test is now : ";

for (int i = 0; i < test.sz(); i++)

cout << test[i] << " ";

cout << endl;

cout << "exam is now : ";

cout << exam << endl;

return 0;

}