Communication Software Design

Lab 09 vector implementation

12/6/2019

In previous lab, we have implemented the vector class (shown below) so that it can be used to store type of string elements (strings only).
And a print() function which can print all elements was added.

```
class vector {
    int sz;     // the size
    double* elem;   // pointer to elements
 public:
    vector(int s) :sz(s), elem(new double[s]);  // constructor
    ~vector();   // destructor , deallocates memory

    double get(int n)  //access: read
    void set(int n, double v);  // set values
    int size() const { return sz; }     // the number of elements
 };
```

```
string NumberToString ( int number ) // #includ <sstream> to use
this
{
    ostringstream ss;
    ss << number;
    return ss.str();
}

int main()
{
        vector v(5);
        v.print();  // this should display -, -, -, -, -
        for (int i=0; i<v.size(); ++i)
        {
            string s = NumberToString(i);
            v.set(i,s);
        }
        v.print();// this should display 0, 1, 2, 3, 4
```

main.cpp

Please continue to modify your “string” vector and implement the **copy constructor** and **copy assignment** using deep copy technique.
In addition, overload the **subscript operator []**, so that each vector element can be access with the syntax: v1[i] as shown in main.cpp below.

```
int main()
{
        vector v1(5);
        cout<<"v1: ";
        v1.print();  // this should display -, -, -, -, -
        for (int i=0; i<v1.size(); ++i) {
            string s = NumberToString(i);
            v1[i]= s;
        }
        vector v2 = v1;   // copy constructor
        v1[0] = "-";      // testing deep/shallow copy
        vector v3(2);
        cout<<"v2: ";
        v2.print();       // this should display 0, 1, 2, 3, 4
        v2[3] = "9";      // testing deep/shallow copy

        vector v3(2);
        v3 = v2;          // copy assignment
        v2[2] = "-";      // testing deep/shallow copy
        cout<<"v3: ";
        v3.print();       // this should display 0, 1, 2, 9, 4
        cout<<"v1: ";
        v1.print();       // this should display -, 1, 2, 3, 4
```