PROGRAMMING I – HELPER INSTRUCTIONS

ACKNOWLEDGEMENT: THE SLIDES ARE PREPARED FROM SLIDES PROVIDED BY NANCY M. AMATO AND JORY DENNY
OVERVIEW

- Implement vector, list and stack using vector and list as underlying container.
- Interface is modeled after Standard Template Library.
- Reference use: [http://www.cplusplus.com/](http://www.cplusplus.com/) to know more about what each function behaves
- Use Prog01/Docs/html/index.html to find the list of things you need to do and other instructions
COMPILING AND RUNNING YOUR CODE

- Change directory to where your code lies: cd Prog01
- Compile your code: make
- Test your code: ./test_vector.o
SUGGESTED PLAN

- Start with vector.h
- Complete default and copy constructor
- Assignment operator
- reserve, push_back, push_back_incremental, pop_back, resize
 Constructors are the member functions that initializes an object for a class

- Default constructor: No initialization information
- Parameterized constructor: Initialization information passed as function parameters
- Copy Constructor: Object of same the class is passed as parameter to the constructor

```cpp
class Point{
    int x;
    int y;
public:
    Point(); //default constructor
    Point(int _a, int _b); //Parameterized Constructor
    Point(const Point& _p); //Copy constructor
}
```

**Exercise:** Which constructor invoked?
- Point A;
- Point B(A);
- Point C(10,15);
Default function (constructor) parameters: The members are assigned the default values when not passed as parameter in the function call.

- Point(int x=0, int y=0);
- Point A; //creates a Point object with x=0, y=0
- Point B(10); //creates a Point object with x=10 and y=0
- Point C(10,10); //creates a Point object with x=10 and y=10
- Note the default parameters must be set from rightmost parameter
  - Point(int x, int y=0); //allowed
  - Point(int x=0, int y); //not allowed

Initializer list: Invoking the constructor of the member variables before the body of constructor being executed

Point(int _a, int _b) : x(_a), y(_b) { …}
Arrays, Pointers and Dynamic Allocation (CH 1.1.3)

- Pointer is a variable that holds the address of another data
  - & - address operator; * - dereferencing a pointer
  - int x = 10;
  - int* y = &x; //y is a pointer containing address of x
  - cout<<*y; //output 10
- Name of the array is pointer to first element in the array
  - int arr[10];
  - *(arr+4); //same as arr[4]

- Dynamic allocation and new operator:
  - new allocates memory for an object, calls the objects constructor and returns a pointer to the object
    - Point* A = new Point(10,20);
    - Dereference a member of pointer variable using ->
      - cout<<A->x; //output 10
- Can be used to allocate or create an array of dynamic size.
  - int *arr = new int[10];
DESTRUCTORS

- Invoked when an object cease to exist.
- What it should contain? Releasing of resources (i.e., memory)
- delete operator helps in deallocating memory
  - delete [] arr; // deletes dynamically allocated array
  - delete A; // deletes a pointer