Dynamic Memory Allocation

Dynamic Memory Allocation with Operators new and delete

- **new** and **delete**
  - Used for dynamic memory allocation
    - Superior to C's **malloc** and **free**
  - **new**
    - Creates an object of the proper size, calls its constructor and returns a pointer of the correct type
  - **delete**
    - Destroys object and frees space
  - Examples of **new**

```
TypeName *typeNamePtr;
```

    - Creates pointer to a **TypeName** object

```
typeNamePtr = new TypeName;
```

    - **new** creates **TypeName** object, returns pointer (which **typeNamePtr** is set equal to)

Dynamic Memory Allocation with Operators new and delete

  - Examples of **delete**

```
delete typeNamePtr;
```

    - Calls destructor for **TypeName** object and frees memory

```
Delete [] arrayPtr;
```

    - Used to dynamically delete an array
- Initializing objects

```
double *thingPtr = new double(
 3.14159 );
```

  - Initializes object of type **double** to **3.14159**

```
int *arrayPtr = new int[ 10 ];
```

  - Creates a ten element **int** array and assigns it to **arrayPtr**