Arrays (Review)

```cpp
#include <iostream>
#include <stdlib>
using namespace std;

int main()
{
    const int MAX_NUM=5;
    int i;
    int numbers[MAX_NUM];
    for( i = 0; i < MAX_NUM; i++ )
    {
        numbers[i] = 0;
    }
    for( i = 0; i < MAX_NUM; i++ )
    {
        cout << "The number at " << i << " is: " << numbers[i] << endl;
    }
    return 0;
}
```
Arrays (Review)

```cpp
#include <iostream>
#include <stdlib>
using namespace std;

int main()
{
    const int MAX_NUM=5;
    int i;
    int numbers[MAX_NUM];
    for( i = 0; i < MAX_NUM; i++ )
    {
        numbers[i] = 0;
    }
    for( i = 0; i < MAX_NUM; i++ )
    {
        cout << "The number at " << i << " is: "
             << numbers[i] << endl;
    }
    return 0;
}
```

All elements in the array are of the same type
Array is of a fixed size
Elements indexed from 0 to (size of array – 1)

Vectors

Different from arrays!
- Vectors are not fixed size. Can grow and shrink
- Part of the Standard Template Library (STL) [more about this in the future]
- Uses template notation [more about this in the future]
Vectors

1. Declaring a vector:

```cpp
vector<int> v;
```
- Creates a vector object that is empty (calls the default constructor)
- `<int>` is template notation indicating that the vector elements are ints (we say the base type is an int)

2. Setting an element

```cpp
v[i] = 42;
```
- Use square brackets to index into vector
- Vectors index from 0 to (size of vector – 1)
- **Note:** ith element must already exist
3. Adding an element

```cpp
v.push_back(100);
v.push_back(200);
v.push_back(300);
```

- Adds elements to the end of the vector
- ie. 100 is at index 0, 200 is at index 1, 300 is at index 2

4. Getting the size of a Vector

```cpp
v.size();
```
- `v.size()` returns the number of elements in vector v
- `v.capacity()` returns the number of elements that the vector can store

- size <= capacity

<table>
<thead>
<tr>
<th>100</th>
<th>200</th>
<th>300</th>
</tr>
</thead>
</table>

Eg. size is 3, capacity is 5
Vectors

- What happens when the capacity is full and you add another element?
- C++ automatically creates a bigger capacity and copies the elements over
- Usually doubles in size

```
100 200 300 400 500
```

Vectors

- `v.reserve(25)` explicitly increases the capacity of the vector v to be 25
- `v.resize(10)` resizes the vector to size 10. If the vector originally had more than 10 elements, the new vector stores only the first 10 elements.
#include <iostream>
#include <cstdlib>
#include <vector>
using namespace std;

int main()
{
    int i;
    const int MAX_NUM = 5;
    vector<int> numbers;

    for(i = 0; i < MAX_NUM; i++)
    {
        numbers.push_back(rand());
    }

    for(i = 0; i < numbers.size(); i++)
    {
        cout << "The number at " << i << " is: " << numbers[i] << endl;
    }

    return 0;
}