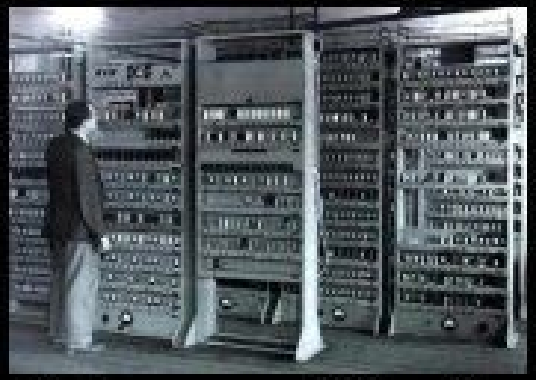
Computer Science Major

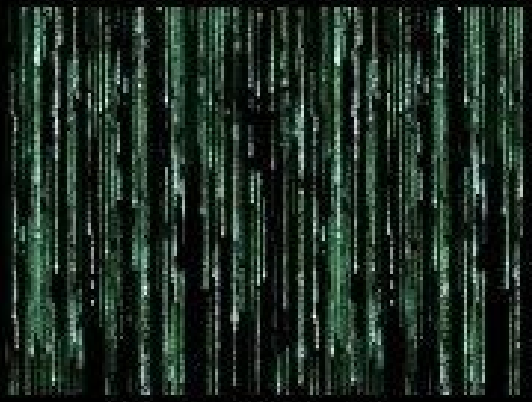
What my friends think I do.



What my mom thinks I do.

What society thinks I do.

What my professor thinks I do in class.



What I think I do.

What I actually do.

Arrays

CS2308
Gentry Atkinson

Mathematical Foundation

Algebra

Scalar value:

$x = 1$

Vector value:

$\bar{x} = \langle 1, 2, 3 \rangle$

Vector elements:

$X_0 = 1$

$X_1 = 2$

$X_2 = 3$

C++

Scalar value:

```
int x = 1;
```

Array value:

```
int x[] = {1, 2, 3};
```

Array elements:

```
X[0] = 1;
X[1] = 2;
X[2] = 3;
```

Arrays in C++

- **Store collections of same-type values.**
- **Must be given a fixed size at declaration.**
- **Values are stored contiguously in memory.**
- **Indexes are used to access individual elements of the array.**
- **The first valid index is 0!**

Example 1

```
int main(int argc, char** argv){
    const int SIZE = 3;
    int a[SIZE];
    for(int i = 0; i < SIZE; i++)
        a[i] = i+1;
    for(int i = SIZE-1; i >=0; i--)
        cout << a[i] << endl;
    return 0;
} //try to predict the output
```

Out of Bounds

- Arrays are given a certain amount of memory when they are declared.
- The compiler trusts us to not access arrays outside of their legal memory.
- Reading an array out of bounds will fetch garbage values.
- Writing an array out of bounds will crash a program.

Example 2

```
int main(int argc, char** argv){
    const int SIZE = 3;
    int a[SIZE] = {1, 2, 3};
    for(int i = 1; i <= SIZE; i++){
        cout << "a [" << i << "] is " << a[i];
        cout << endl;
    }
    return 0;
} //try to predict the output
```

Multi-dimensional Arrays

- An array can be given one size at declaration or several.
- A 1-D array is like a list. A 2-D array is like a table.
- Elements in a 2-D array are referenced using two indexes: array[row][column].
- Too many dimensions can get confusing.

Example 3

```
int main(int argc, char** argv){
    const int SIZE = 3;
    int a[SIZE][SIZE] = { {1, 2, 3}, {2, 3, 4},{4, 5, 6}};
    for(int row = 0; row < SIZE; row++){
        for(int col = 0; col < SIZE; col++)
            cout << a[row][col] << ' ';
        cout << endl;
    }
} //try to predict the output
```

For an array a[SIZE]

The first index is a[0]

The last index is a[SIZE-1]

Questions or Comments?