Intro to Arrays

Arrays: Common Features

With all languages in the C/C++/Java/C# family, an array

- "Looks like" this:

- Has a fixed size, determined at creation time
- Stores same-type items
- Has indices 0 to (size - 1) accessed with the [] operator, e.g. data[i]
- May be multidimensional (e.g. 1D, 2D, 3D, ...)
- May be initialized like this: data = {18, 6, 3, 9, 12};
- Has fast "random access"
- Is not built for dynamic growth. Solutions for "full" scenarios: create another array twice the size and copy the data. Or use a circular array. Or use a different data structure such as a linked list instead.

Differences: Index Check

C/C++ arrays differ from the more modern Java/C# arrays like so:

- Array indices are not checked!
  E.g. data[num] compiles and runs regardless of the value of num.
  In Java an exception would be thrown, but C has no exceptions.
  THE BAD NEWS:

  - Program may crash with a segmentation error
  - Or:
    - if writing, memory is corrupted
    - if reading, bogus data is read

WHAT'S THE GOOD NEWS????

Differences: Declaration

- In Java, arrays are objects. Array creation is a two step process: declare the reference variable; create the object. Commonly both are done in one statement. A for-loop usually uses the 'length' field.

```
int[] data;
data = new data[10];    // int[] data = new data[10];
for (int i = 0; i < data.length; i++)
    // do something with data[i]
```

- In C, arrays are not objects. They are declared like this:

```
int data[10];
```

- Because C arrays are not objects, there is no 'length' field. Recommendation: Store the length in a variable. Why???

```
const int SIZE = 10     // or #define SIZE 10
int i = 0;
int data[SIZE];
for (i = 0; i < SIZE; i++)
    // do something with data[i]
```

- Function Parameter Declaration:

```
void foo(int data[], int size);    // C
void foo(int[] data);    // Java
```

Differences: Addressing

In C, unlike Java:

- Array items can be accessed directly via their address, e.g.

```
scanf("%d", &data[3]);  // read an int; store it in the 4th spot
```

- The value of an array variable IS the actual virtual address of the array (whereas Java uses references and has another level of indirection).

- Arithmetic operations with array variables are allowed, like so:

```
scanf("%d", data + 3);  // read an int; store it in the 4th spot
```

- You can see how your stack grows:

```
int dataX[size];
int dataY[size];

printf("dataX is %u, dataY is %u\n", dataX, dataY);
printf("dataX is %x, dataY is %x\n", dataX, dataY);
```

Differences: Strings

- In Java Strings are objects
  String s = new String("hello");

h	e	l	l	o	\0

- C has neither objects nor a String datatype
- In C, strings are stored as a null-terminated sequence of characters
- They can be declared in a few ways:

```
char data = "hello";

char line[6];
line[0] = 'h';
...
line[5] = '\0';
```