(1) Overview

- More on pointers and memory:
  - Returning a pointer
  - Stack memory
  - Heap memory

(2) Side-by-side function parameters in C++

Given the following function declaration, complete main() by declaring a set of variables and ca call to foo() such that the code compiles correctly.

```cpp
void foo(int a, int &b, int *c, int d[]);

int main(){

    int x = 0, y = 0, *z, arr[];

    foo(x, y, z, arr);

    cout << x << ", " << y << endl;

    return 0;
}
```

(3) Returning a pointer?

findMax

Can a function return a pointer? Hey, I was told I can

Consider the following snippet:

```cpp
int * foo(){
    int * x;
    return x;
}
```
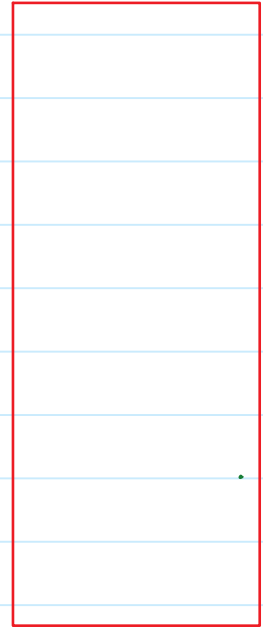
Yes, this is allowed.

Wait, I was told we CANNOT return arrays in C++. But if a pointer is an array..

```cpp
int * foo(){
    int x[] = {7,10,21,4,5,6,7,8,9};
    return x;
}
```
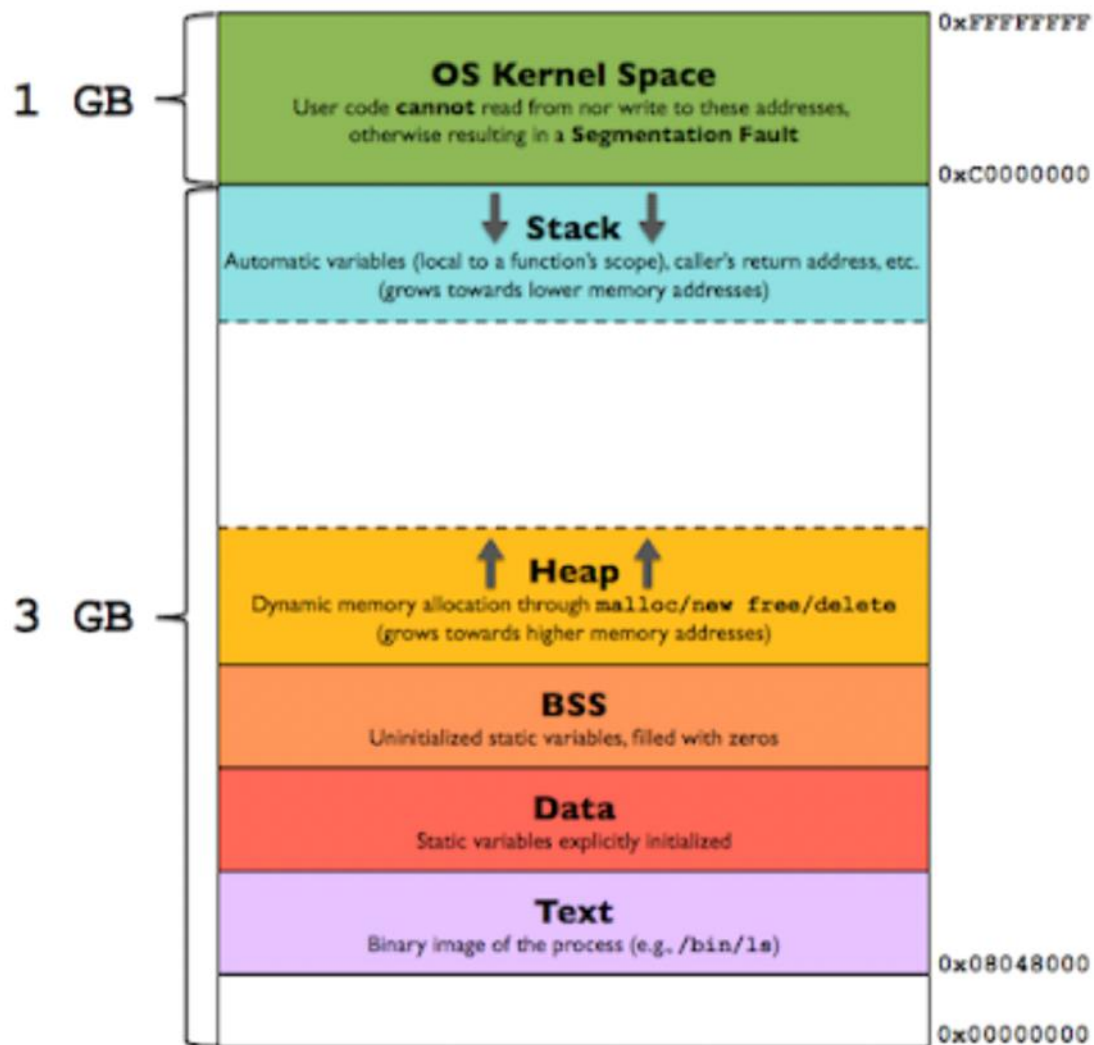
(4) The Stack and automatic variables

```c
int foo(){
    int a = 7;
    return a;
}
int main(){
    int x[] = {7,10,21,4,5,6,7,8,9};
    x[3] = foo();
    return 0;
}
```

Program flow:

(5) Memory Organization: Stack and Heap

source: A. Trivedi

(6) The Heap: Dynamic Memory

Let's talk about the other type of memory: the heap.

In order to use the heap in C++ we use **new** and **delete.**

Creating variables on the heap is a bit different than variables on the stack. The variable doesn't get an actual name like on the stack. Instead, we use a pointer to allocate the memory for the variable.

Example of using an integer variable on the heap:

```
// First we instantiate a pointer
// The pointer variable is ON THE STACK!
int *p1;

// Then what?
```

(7) Dynamically Allocated Array

We can declare arrays on the heap as well. For example, let's make an array of length 5:

```
int *ptr;
ptr = new int[5];
```

(8) Example with an array on the heap

- Assume we have a program that is prompting the user to input integer data. We'd like to store the integers in an array.
- We want to allow the user to keep entering as many integers at they would like.
- How to set the array length?

Say, we have an array of length 3. Our program is reading in from the user, and after reading 3 items, it fills up the *original* array.