First-class Functions

first-class entity => put in /
get from variable j => pass to
get back from function
1st class entities in C

bools, ints, floats, chars, etc.

(strings), arrays

In other languages:

functions, threads (processes), packages (modules)
Abstraction is fundamental to programming

```c
int f1()
{
    return global var + 7;
}

int f2()
{
    return global var + 7;
}
```

```c
int f(int offset)
{
    return global var + offset;
}
```
function sum(int a[]) {
    int total = 0;
    for (int i = 0; i < a.length; i++) {
        total += a[i];
    }
    return total;
}

function product(int a[]) {
    int prod = 1;
    for (int i = 0; i < a.length; i++) {
        prod *= a[i];
    }
    return prod;
}
function fold(data, initval, f)
    local total = initval
    for i = 0, #data, 1 do
        total = f(total, data[i])
    end
    return total
function sum(data) {
    return fold(data, 0, "+")
}

function product(data) {
    return fold(data, 1, "*")
}
So what? In C, functions aren't first-class.

Well, we can sort of fake it. C has function pointers.
Well when code is compiled,

0 \xrightarrow{\text{global data}} \text{start address}

\text{\$1000} \xleftarrow{f_1}

\text{\$78} \xleftarrow{f_2}

\text{\$78} \xleftarrow{f_3}
How to declare function ptr?

```c
int (*func)(int, int) fp;
int min (int x, int y) fp now points to the min value
```
\[
\text{int val} = (*\text{fp})(a, b);
\]

\[\text{invokes function pointed to by fp} \]

\[
\text{if \ (user-input = \ "min\") } \{ \text{fp} = \&\text{min}; \}
\]

\[
\text{else } \{ \text{fp} = \&\text{max}; \}
\]

\[\text{print(val);} \]
In Java

interface function \{ 
    int invoke(int a, int b);
\}

class Sum implements function \{ 
    public int invoke(int a, int b) \{ 
        return a + b;
    \}
\}
```java
PSV man (Strings <1rs) ?

// Create a vector of data

Sum s = new Sum();
int total = 0;
for (i = 0; i < data.length; i++)

```
printf("The answer is \%d \n", value);

printf("%d + %d = %d \n", 5, 4, 5+4);

printf("My name is \%s. I am \%d years old and I have \%d kids and \%d cars and \%d boats.", Name, n. years, n-3 kids, n-2 cars, n. boats).
printf takes a variable # of arguments

_vararg function

scanf — also _vararg

#include <varargs.h>
There's lots more to see in C

- insert assembly code
- dynamic loading of libraries
- ffi (foreign function interface)

JNI
Not even mentioning

C#

.NET  C#  IronPython  CLR  F#
Quiz

1. What is memory management?
   Describe

2. Name 3 differences between C & Java.