C Function Pointers

CS 2060

Prof. Jonathan Ventura

Function Pointers

- In C, we can also obtain a pointer to a *function*.
- As always, the pointer is a memory address – the address of the function in memory.

```
void (*myfn)( int arg1, int arg2 );
```

```
void (*myfn)( int, int );
```

- Function pointers are useful for making customizable code – the end user of your library can provide a custom function using a function pointer.

Function Pointers

```
void runTask( void (*callbackFn)( int status ) )
{
    // .. do some computation

    (*callbackFn)( 0 );
}
```

The `runTask` function takes a function pointer as an argument. At the end of the function, it calls the provided callback function.

Function Pointers

```
// compare returns 1 if a should be before b
void isSorted( const int array[], size_t size,
               int (*compare)( int a, int b ) )
{
    for ( size_t i = 0; i < size-1; i++ )
    {
        if ( (*compare)( array[i], array[i+1] ) == 0 ) return 0;
    }

    return 1;
}
```

Function Pointers

```
int compareAscending( int a, int b )
{
    return ( a <= b );
}

int compareDescending( int a, int b )
{
    return ( b >= a );
}
```

Function Pointers

```
void swap( int array[], size_t i, size_t j )
{
    int save = array[i];
    array[i] = array[j];
    array[j] = save;
}

void shuffle( int array[], size_t size )
{
    for ( size_t i = 0; i < size; i++ ) swap( array, i, rand() % size )
}

void bogoSort( int array[], size_t size,
               int (*compare)( int a, int b ) )
{
    while ( !isSorted( array, size, compare ) ) shuffle( array, size );
}
```