Announcements

• Gradescope is up and running for all problems in the homework and extra credit!

• For homework 1, problem 2, check your score. If it is 55, please resubmit your program
  • The number of points was added incorrectly

• Be sure you use this command to run your program in the CSIF before submitting it to Gradescope:
  
gcc -ansi -pedantic -Wall filename.c -o filename

• Some compilers allow // to comment out the rest of the line
  • Not part of the C99 standard
Functions

- Perform some task the program will do repeatedly
- Helpful for organizing programs
- Improves readability
Format

• Here is a function definition:

```c
int add17(int num){
    int y; /* used to hold sum */
    y = num + 17;
    return(y);
}
```

• Here is a function call:

```c
    . . .
    sum1 = add17(53);
    . . .
    sum2 = add17(-12);
    . . .
```
In Detail – Function Definition

```c
int funct(int par1, float par2, char par3){ . . .
```

- type of what function returns; if it doesn’t return anything, use `void` here
- name of function
- type of first parameter
- variable representing first parameter
- type of second parameter
- variable representing second parameter
- type of third parameter
- variable representing third parameter
In Detail – Function Call

```c
int x;
float fx;
x = funct(7, fx, 'a');
```

- Arguments are matched with parameters in order
- Here, from previous slide:
  - par1 is 7
  - par2 is the value contained in fx
  - par3 is ‘a’
- Note parameter types matches argument types
Prototypes or Forward Declarations

• Functions must be declared before use
• If defined before use, the function type, name, and parameter list serves as the declaration
• If defined after use, compiler makes assumptions about the types of parameters and function
  • And gcc will give you a warning
• A function prototype looks exactly like the first line of a function definition
  • int funct(int par1, float par2, char par3);
  • Note the “;” at the end!
void swap(int a, int b) {
    int tmp;
    temp = a;
    a = b;
    b = temp;
}
a = 13;
b = 5;
printf("a = %d, b = %d\n", a, b);
swap(a, b);
printf("a = %d, b = %d\n", a, b);
The Stack

void swap(int a, int b)
{
    int tmp;
    temp = a;
    a = b;
    b = temp;
}

. . .
x = 3; y = 13;
swar(x, u);
. . .
print("x = %d; y = %d\n", x, y);
Scope

• When multiple variables have the same name, which one is used?
  • Rule #1: two variables cannot have the same name in a block (e.g., function)
• Use the variable that is “nearest” to the reference
  • That’s the one in scope