Homework 2

Parth Parth

02/24/2022

1. Consider the following C code  
  
#include <stdio.h>  
#include <stdlib.h>  
  
float f;  
  
void func( float f ) {  
     printf( "f at mark1 %f\n", f );  
     f = 3.0;  // f1  
     printf( "f at mark2 %f\n", f );  
}  
  
int main( int argc, char \*argv[]) {  
     f = 2.0; // f2  
     printf( "f at mark0 %f\n", f );  
     func( f );  
     printf( "f at mark4 %f\n", f );  
}  
  
a. In the assignment statement with // f1 next to it, what variable f  
does that assignment refer to? A file-level variable or a function-level  
variable?

It refers to the function-level f  
  
b. In the assignment statement with // f2 next to it, what variable f  
does that assignment refer to? A file-level variable or a function-level  
variable?

File-level variable  
  
c. What is printed when this code is executed? (obviously, you can run  
the code, but the idea here is to be able to predict it first).

f at mark0 2.0

f at mark1 2.0

f at mark2 3.0

f at mark4 2.0  
  
2. Write your own C code example of shadowing

float a=4.0;

void fxn() {

float a=5.0; // this a will shadow the a declared in file-level scope

}

3. Write your own C code example of a variable whose lifetime is not  
linked to its scope.

int \*ptr=(int \*) malloc(sizeof(int));  
  
4. Identify two differences between scoping rules in C and Python.

i. Python has no block level scope while C does.

ii. In Python, to manipulate a global variable (file-scope), it needs to be declared as “global variableName” inside functions. While in C, unless the variable is shadowed, it can automatically be manipulated inside a function.  
  
5. Identify two differences between scoping rules in C and Java.

i. Java does not allow shadowing of variables while C does

ii. Since Java is Object oriented, you can refer to the class members in Java using ClassName.memberName or this.memberName. Since C is not object oriented, data members can only be accessed by name. Since Java does not shadow, variables inside a function and class-level variables can both be accessed.