COMP20005 Workshop Week 6

1	Scopes, *exercise 6.2*
2	Pointers, Pointers as Function Arguments, *exercise 6.5, discuss exercise 6.9*
	Preparation for Assignment 1
L	• ***Redirection***
A	
B	***6.9, triangle.c, other ex from C05 and C06***

Scopes: local variables

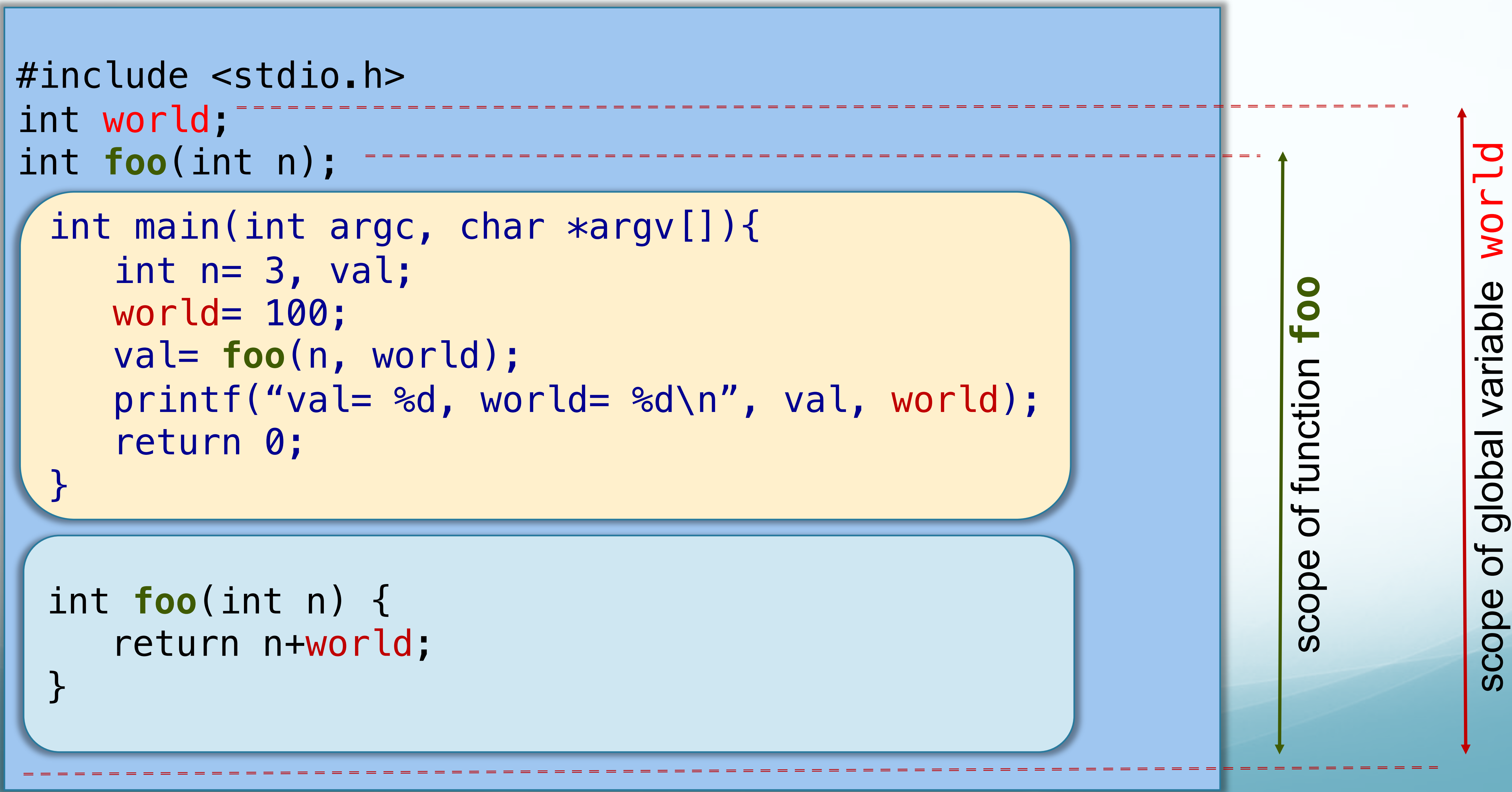
```
#include <stdio.h>
int fact(int n);
int main(int argc, char *argv[]){
    int n= 3, val;
    val= fact(n);
    printf("%d! = %d\n", n, val);
    return 0;
}

int fact(int n) {
    int i, f= 1;
    for (i=1; i<=n; i++) {
        f *= i;
    }
    return f;
}
```

`argc`, `argv`, `n`, and `val` available here

`n`, `i`, and `f` available here

Scopes: global objects



```c
#include <stdio.h>
int world;
int foo(int n);
int main(int argc, char *argv[]){
    int n= 3, val;
    world= 100;
    val= foo(n, world);
    printf("val= %d, world= %d\n", val, world);
    return 0;
}

int foo(int n) {
    return n+world;
}
```

A Rule: Never use global variables

```
#include <stdio.h>
~~int world;~~
int foo(int n);
int main(int argc, char *argv[]){
   int n= 3, val;
   int world= 100;
   val= foo(n, world);
   printf(“val= %d, world= %d\n”, val, world);
   return 0;
}

int foo(int n, int world) {
   return n + world;
}
```

6.02: *For each of the 3 marked points, write down a list of all of the program-declared variables and functions that are in scope at that point, and for each identifier, its type.*

```c
int bill(int jack, int jane);
double jane(double dick, int fred, double dave);

int trev;

int main(int argc, char *argv[]) {
     double beth;
     int pete, bill;     /* -- point #1 -- */
     return 0;
}

int bill (int jack, int jane) {
     int mary;
     double zack;        /* -- point #2 -- */
     return 0;
}

double jane(double dick, int fred, double dave) {
     double trev;        /* -- point #3 -- */
     return 0.0;
}
```

Quiz 1

In executing the program:

```
int a=100, b=200;
void f(int a) {
   a++;
   print("1: a= %d b= %d\n", a, b) ;
}
int main(int argc, char *argv[]) {
  int a=5, b= 10;
  f(a);
  print("2: a= %d b= %d\n", a, b) ;
  return 0;
}
```

what will be printed out?:

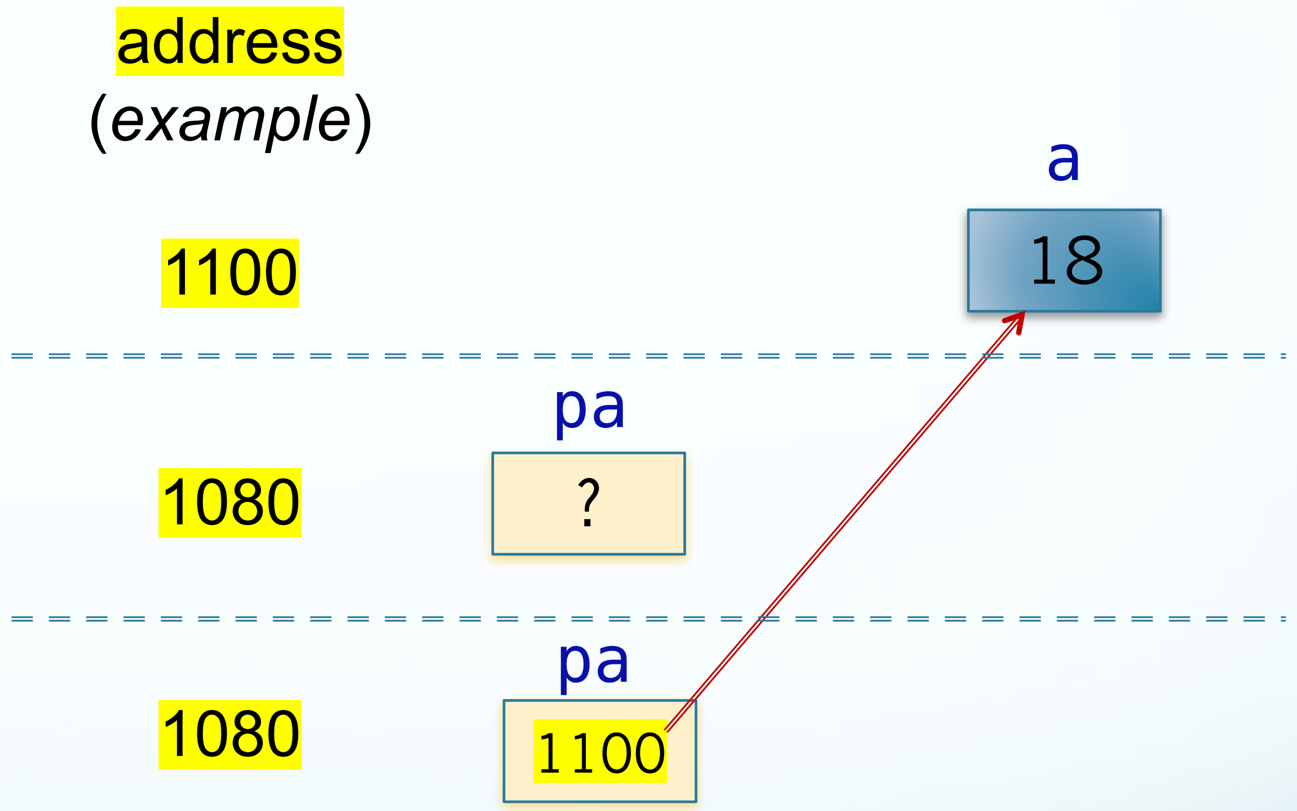
A	1: a= 6 b= 200 2: a= 5 b= 10	B	1: a= 6 b= 200 2: a= 6 b= 10
C	1: a= 6 b= 10 2: a= 5 b= 10	D	1: a= 6 b= 10 2: a= 6 b= 10

Pointers: check your understanding

```
int a= 18;

int *pa;

pa= &a;
```



```
// What is the value of a and pa after:
*pa – (*pa) + 1;
```

unary operators & and * : referencing and dereferencing

```
        int n= 10;
        int *pn;

        pn= &n;
```

Check your understanding:

a) The datatype of pn is ______

b) If n is at the address 4444, then pn has the value of ______

c) The value of *pn is ______

d) After

```
        *pn= 100;
```

the value of pn is __________, of n is ___________

e) What is the effect of:

```
                    *(&n) = 1;
```

Pointers – application in function parameters

1	`int n=10;`	
2	`printf(“%d”, n);`	What sent to `printf` ? Can `printf` change the value of `n`?
3	`scanf(“%d”, &n);`	What sent to `scanf`? What `scanf` do to `&n`, to `n`?
4	`swap(&n, &m);`	What passed to `swap`? Can this call make change to `&n` or `&m`? Can this call make change to `m` or `n`?

5	`void int_swap(int *a, int *b){`
	`    ???`
	`}`	

pointers as function parameters: an example

Using pointers as parameters, function can have multiple output!

Example: Line 4 leads to the change of **sum** and **product**.

→ function sAndP effectively has 2 outputs!

```
1   int main(...) {
2     int a=2, b=4, sum, product;
3        ...
4     sAndP(a, b, &sum, &product);
5
6     printf("sum=%d",                     sum);
7     printf("prod=%d”,                    product);
8       ...
9   }

11  void sAndP(int m, int n, int *ps, int *pp ) {

12    *ps = m + n ;
13
      *pp = m * n ;

14
    }
```

Quiz 2

After executing the fragment:

```
int x= 10;
f(&x);
printf(“x= %d\n”, x);
```

the output is:

```
x= 0
```

which function has been used in the call `f(&x)` *?*

A:

```
int f(int n) {
    return 0;
}
```

B:

```
void f( int *n) {
   &n= 0;
}
```

C:

```
void f (int *n) {
    n= 0;
}
```

D:

```
void f( int *n) {
   *n= 0;
}
```

Quiz 3

Given function:

```
void f(int a, int *b) {
   a= 1;
   *b = 2;
}
```

Assuming the following fragment is in a valid main(). What will be printed out?

```
int m= 5;
int n= 10;
f(m, &n);
printf(“m= %d, n= %d\n”, m, n);
```

A) m= 5, n= 10	B) m= 1, n= 2
C) m= 5, n= 2	D) m=1, n= 10

Discuss & DoTogether in grok: Exercise 6.05

Write a function
`int_sort2`
that orders its two arguments so that the numerically smaller value is placed into the underlying variable corresponding to the first pointer argument, and the larger into the variable corresponding to the second argument pointer.

```
int main(int argc, char *argv[]) {
    ...

    printf("Before: v1 = %d, v2 = %d\n", v1, v2);
    int_sort2(&v1, &v2);
    printf("After:  v1 = %d, v2 = %d\n", v1, v2);

    return 0;
}

void int_sort2(int *x1, int *x2) {


}
```

Demonstration: Stuffs for Assignment

- Redirection (with 6.05 or a program for copying files)

Design: Ex 6.09

Lab & Notes

- Implement 6.9 (and Re-implement 6.5 if still in doubt)
- *Do the exercise with* [triangle.c]() *as described in LMS Week 6 Workshop Content*
- implement not-yet-done Exercises in grok C05 and C06

Assignment 1 released Wed next week!

- Do as much as you can by Week 7 Workshop
- Try to submit a few times
- Regularly use Discussion Forum
- Q&A in Week 7 Workshop

Have a Great Easter Break!