Lecture #4b

Pointers and Functions

NUS | School of
National University | Computing
of Singapore

Questions?

Ask at https://app.sli.do/event/bRPtUxgykAQjjF5XBpLedo

OR

Scan and ask your questions here!
(May be obscured in some slides)

1.7 Tracing Pointers (1/2)

- Trace the code below manually to obtain the outputs.
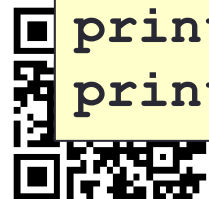- Compare your outputs with your neighbours.

TracePointers.c

```c
int a = 8, b = 15, c = 23;
int *p1, *p2, *p3;

p1 = &b;
p2 = &c;
p3 = p2;
printf("1: %d %d %d\n", *p1, *p2, *p3);

*p1 *= a;
while (*p2 > 0) {
  *p2 -= a;
  (*p1)++;
}
printf("2: %d %d %d\n", *p1, *p2, *p3);
printf("3: %d %d %d\n", a, b, c);
```
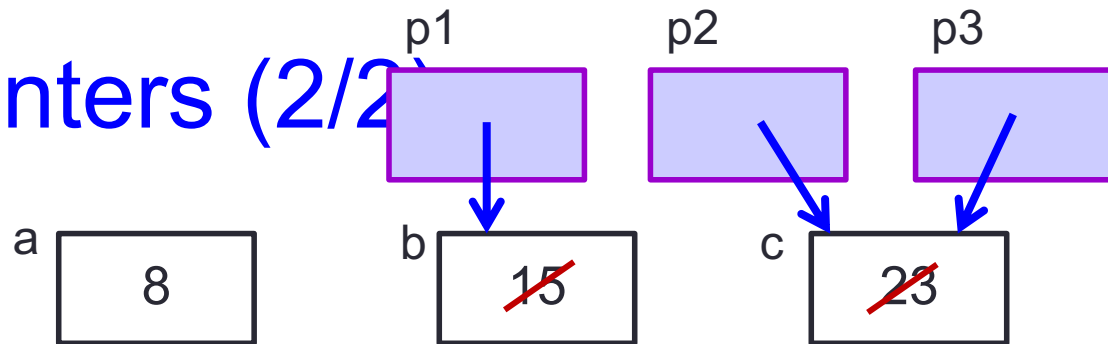
1.7 Tracing Pointers (2/2)

p1   p2   p3

a 8   b ~~15~~   c ~~23~~

120
~~121~~
~~122~~
123

~~15~~
~~7~~
-1

```
int a = 8, b = 15, c = 23;
int *p1, *p2, *p3;

p1 = &b;
p2 = &c;
p3 = p2;
printf("1: %d %d %d\n", *p1, *p2, *p3);

*p1 *= a;
while (*p2 > 0) {
    *p2 -= a;
    (*p1)++;
}
printf("2: %d %d %d\n", *p1, *p2, *p3);
printf("3: %d %d %d\n", a, b, c);
```

1: 15 23 23

2: 123 -1 -1

3: 8 123 -1

1.8 Incrementing a Pointer

- If p is a pointer variable, what does p = p + 1 (or p++) mean?

Recall Lect#2a slide 16:
int takes up 4 bytes
float takes up 4 bytes
char takes up 1 byte
double takes up 8 bytes

```
int a; float b; char c; double d;
int *ap; float *bp;
char *cp; double *dp;

ap = &a; bp = &b; cp = &c; dp = &d;
printf("%p %p %p %p\n", ap, bp, cp, dp);
```

ffbff0a4 ffbff0a0 ffbff09f ffbff090

```
ap++; bp++; cp++; dp++;
printf("%p %p %p %p\n", ap, bp, cp, dp);
```

+4   +4   +1   +8

ffbff0a8 ffbff0a4 ffbff0a0 ffbff098

```
ap += 3;
printf("%p\n", ap);
```
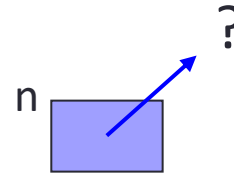
+12

ffbff0b4

IncrementPointers.c

1.9 Common Mistake

CommonMistake.c

```
int *n;

*n = 123;
printf("%d\n", *n);
```

What's wrong with this?
Can you draw the picture?

?

n

- Where is the pointer **n** pointing to?
- Where is the value **123** assigned to?
- Result: Segmentation Fault (core dumped)
  - Remove the file "core" from your directory. It takes up a lot of space!

1.10 Why Do We Use Pointers?

- It might appear that having a pointer to point to a variable is redundant since we can access the variable directly

- The purpose of pointers is apparent later when we pass the address of a variable into a function, for example, in the following scenarios:

  - To pass the addresses of two or more variables to a function so that the function can pass back to its caller new values for the variables

  - To pass the address of the first element of an array to a function so that the function can access all elements in the array

Quiz

- Please complete Pointers and Functions Quiz 1 before 3 pm on 23 August 2022.

End of File