Lecture 9

Wednesday, May 3, 2017 1:40 PM

POINTERS & lab04

Problem Solving with Computers-I

https://ucsb-cs16-wi17.github.io/

C++

```
#include <iostream>
using namespace std;

int main(){
    cout<<"Hola Facebook!";
    return 0;
}
```

GitHub

Wednesday, May 3, 2017 1:40 PM

How comfortable do you feel with using github?

A. Very comfortable in the context of labs, I have a basic understanding of how git works
B. I know how to use it but I have no idea how git works
C. I don't feel comfortable using it
D. I am completely lost

Wednesday, May 3, 2017 1:40 PM

How far along are you with lab04

A. Almost done
B. I am on track to finish
C. I am stuck and don't know how to proceed
D. Haven't started

Wednesday, May 3, 2017 1:40 PM

Swap function – midterm 1

Local variables of a function are created when the function is called. When the function returns these variables are deallocated from memory

```
#include <iostream>
using namespace std;
void swap(int a, int b){
      cout<< "Inside swap"<<endl;
      int tmp = a;
      a = b;
      b = tmp;
      cout<< a << "  " << b<< endl;
}

int main(){
      int x= 10, y=20;
      cout<< "Before swap" <<endl;
      cout<< x<< "  " <<y<<endl;
      swap(x, y);
      cout<< "After swap" <<endl;
      cout<< x<< "  " <<y<<endl;
}
```

swap(x, y); ⇐ Function call

Local variables 'a' & 'b' are created AND contain copies to the values of 'x' & y

These are created when 'swap' is called

a: 10 → 20

b: 20 → 10

Deallocated when function returns

x: 10

y: 20

The function swaps the values of the copies 'a' & 'b' NOT x & y

Wednesday, May 3, 2017 1:40 PM

5

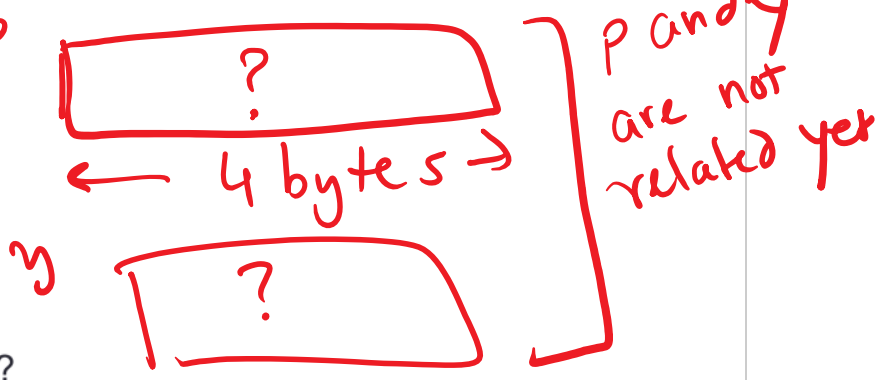
Pointers

- Pointer: A variable that contains the address of another variable
- Declaration: *type* * pointer_name;

```
int *p;
```

//p is a pointer to an int

int y;



How do we initialize a pointer?

6

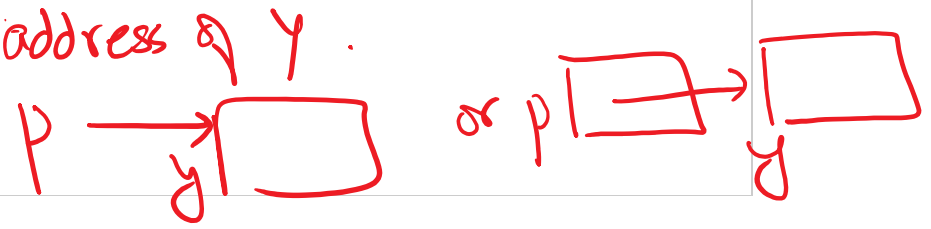
How to make a pointer point to something

```
int *p;
int y;
```

p = &y;

To access the location of a variable, use the address operator '&'

We have a short hand representation for the scenario where p has the address of y.

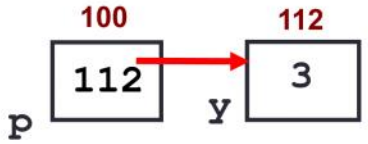


How to make a pointer point to something

```
int *p, y;
```

y = 3;

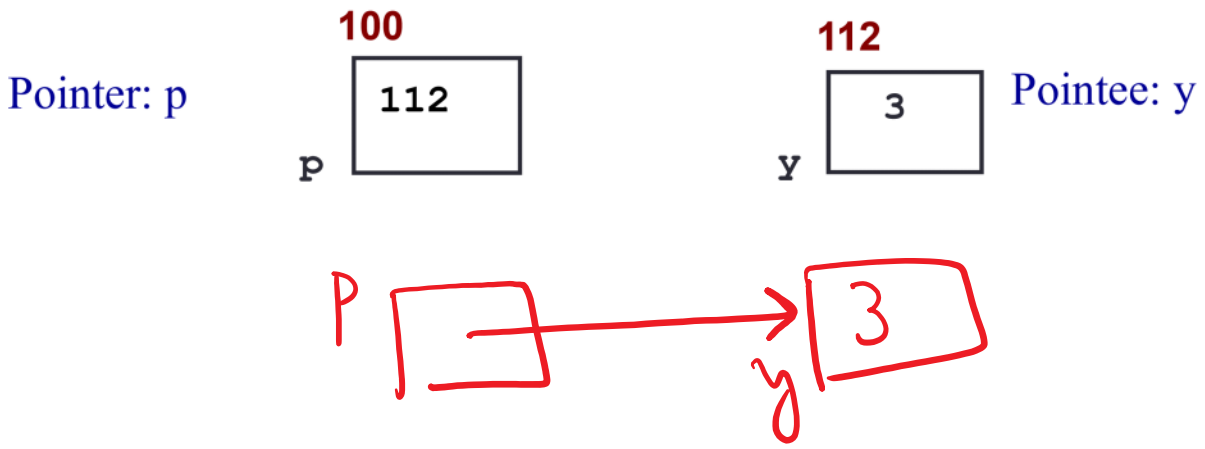
p = &y;



p points to **y**

Wednesday, May 3, 2017 1:40 PM

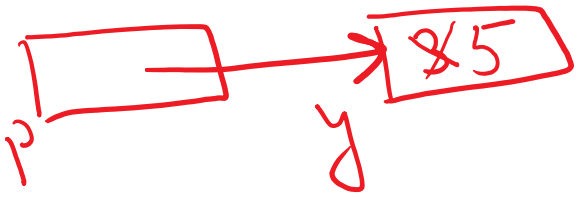
Pointer Diagrams: Diagrams that show the relationship between pointers and pointees



9

You can change the value of a variable using a pointer !

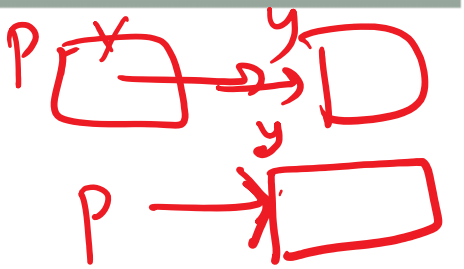
```
int *p, y;
y = 3;
p = &y;

*p = 5;
```



Use dereference * operator to left of pointer name

Wednesday, May 3, 2017 1:41 PM

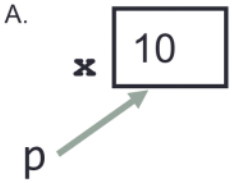
Tracing code involving pointers



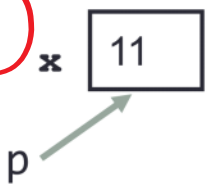

```
int *p, x=10;
p = &x;
*p = *p + 1;
```

Q: Which of the following pointer diagrams best represents the outcome of the above code?

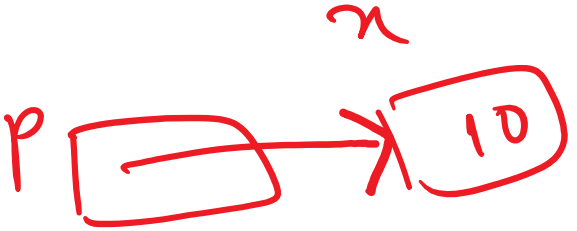
A.



B.



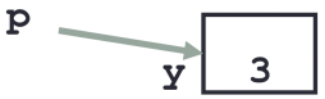
C. Neither, the code is incorrect



Wednesday, May 3, 2017 1:41 PM

11

Two ways of changing the value of a variable



Change the value of y directly: y = 20;

Change the value of y indirectly (via pointer p): *p = 20;

Wednesday, May 3, 2017 1:41 PM

Pointer assignment and pointer arithmetic: Trace the code

```
int x=10, y=20;
int *p1 = &x, *p2 =&y;
p2 = p1;
int **p3;
p3 = &p2;
```

Wednesday, May 3, 2017 1:41 PM

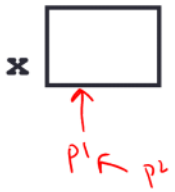
13

Pointer assignment

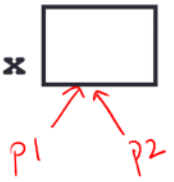
```
int *p1, *p2, x;
p1 = &x;
p2 = p1;
```

Q: Which of the following pointer diagrams best represents the outcome of the above code?

A.



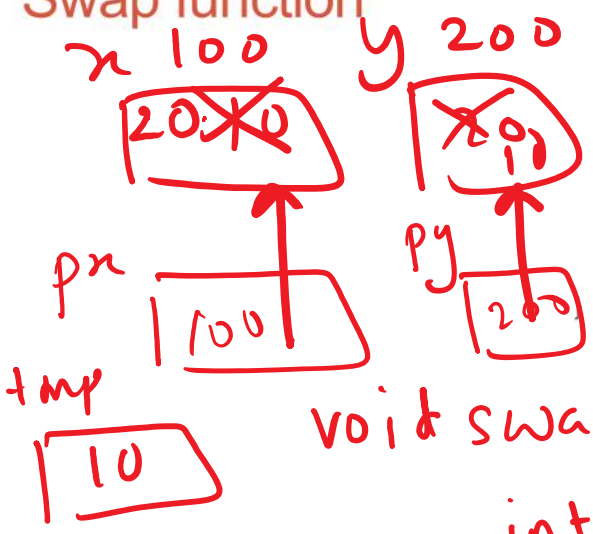
B.



C. Neither, the code is incorrect

Wednesday, May 3, 2017 1:41 PM

Swap function

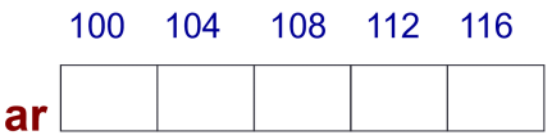



```
int main(){
    int x= 10, y=20;
    cout<< "Before swap" <<endl;
    cout<< x<< "  " <<y<<endl;
    swap(x, y);
    cout<< "After swap" <<endl;
    cout<< x<< "  " <<y<<endl;
```

swap(&x, &y);

```
void swap (int *px, int *py) {
    int tmp = *px;
    *px = *py;
    *py = tmp;
}
```

Arrays and pointers



- `ar` holds the address of the first element (like a pointer)
- `ar[0]` is the same as `*ar`
- Use pointers to pass arrays in functions

```
int ar[5]={65, 85, 97, 75, 95};

int *p;
```

Wednesday, May 3, 2017 1:41 PM

Next time

- What can go wrong when using pointers
- References
- Pointers and structs
- Mechanics of function calls contd.–call by reference