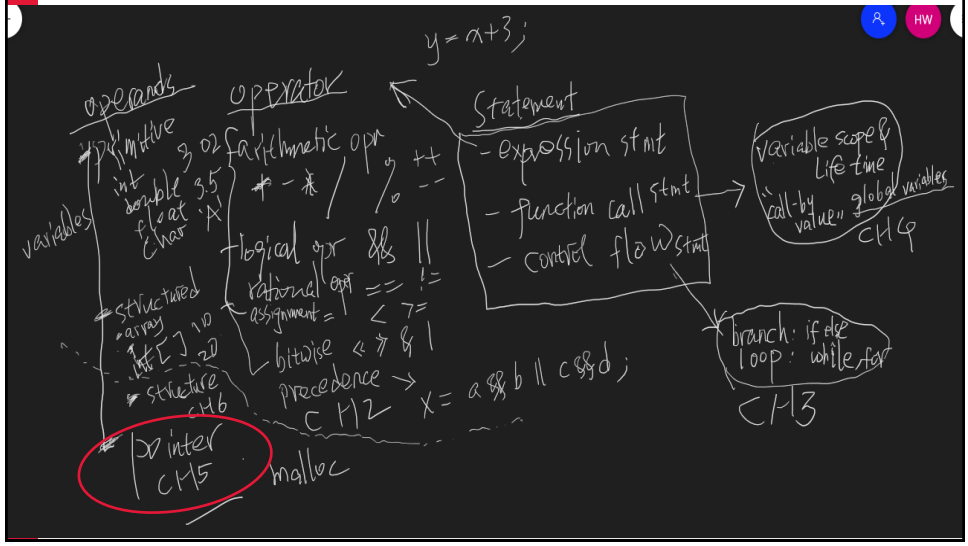
- Now it is time to start POINTERS!!!

1

Roadmap -- How the topics are related



2

Pointers K&R Ch 5

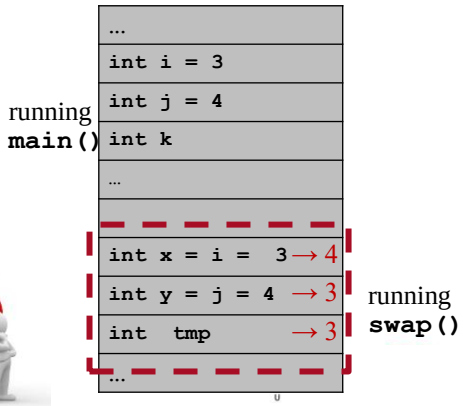
- Basics: Declaration and assignment (5.1)
- Pointer to Pointer (5.6)
- Pointer and functions (5.2)
- Pointer arithmetic (5.4)
- Pointers and arrays (5.3)
- Arrays of pointers (5.6)
- Command line argument (5.10)
- Pointer to arrays and two dimensional arrays (5.9)
- Pointer to functions (5.11)
- Pointer to structures (6.4)
- Memory allocation (extra)

YORK UNIVERSITÉ UNIVERSITY

Motivations: Pass-by-Value

- In C, all functions are **pass by value**
  - Value of the arguments are passed to functions, but not the arguments themselves (i.e., not "~~pass by reference~~")

```
void swap (int x, int y)
{ int tmp;
  tmp = x;
  x = y;
  y = tmp;
}
main(){
  int i=3, j=4;
  swap(i,j)
}
```



```
char fromStr [] = "Hello!";
char toStr [20];

strcpy(toStr, fromStr);     // toStr modified

fgets(toStr, 10, stdin);    // toStr modified
```

- Given an array as an argument, a function can modify the contents of the array -- Arrays are passed *as if* "call-by-reference"
- But isn't C "call-by-value"? -- pass single numerical value
  - How to pass strings to **`strcpy()`**?
  - How does **`strcpy()`**,**`scanf()`**,**`fgets()`** modify argument?
- Also **`scanf ("%d %s", &a, arr); // a arr modified`**
  - Why **`&a`**, why not **`&arr`**

5

5

Mr. Main

Hi, function, I have some manuscripts, stored in lockers (memory), and I want you to repaint them so their color changed.

Ms function

Hi, Mr Main, here is how we function work:
We don't take your original manuscript over the counter (not pass by reference). We always photocopy things passed here (pass by value) .

Mr. Main

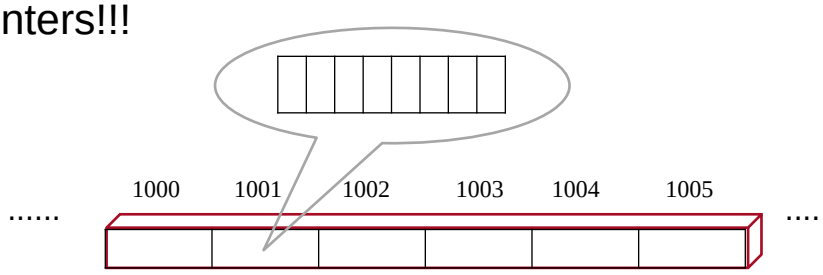
Then, is there a way to have my original manuscripts' color changed by you?

Ms function

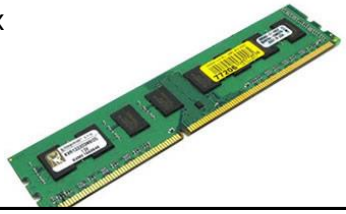
Of course there is a way.
Don't give us the paper.
Instead, give us your ….

7

Pointers!!!

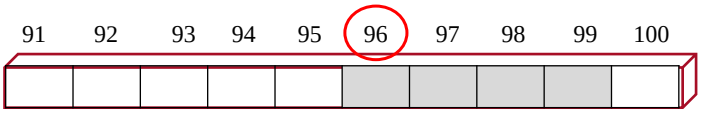


- computers memory
  - Thousands of sequential storage location byte (8 bits)
  - Each byte has a unique address
  - Range 0 ~ max



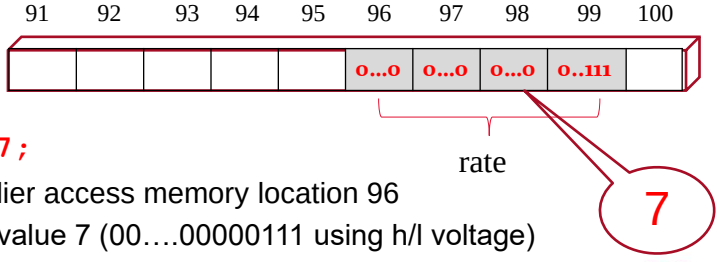
8

8




```
int rate;
```

- set aside memory (4 bytes)
- associates 96 (starting address) with **`rate`**;




```
rate = 7;
```

- Complier access memory location 96
- Store value 7 (00….00000111 using h/l voltage)
- Hidden from us

9

9

C allows us to access and store the addresses of variables

Not in Java

`&x`

- address of a variable, array element. (No expression)

```
&x    &rate
&arr[0]; // later
scanf("%d %d",  &a, &b);
```

`type * p ;`

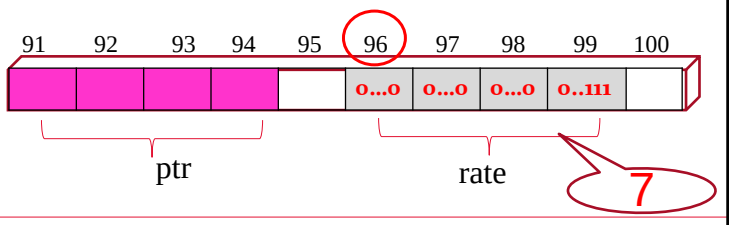
- `p` is a pointer variable capable of storing the address of a int variable -- pointing to variable of type **`type`**

```
int * p, *q;
double * pd;
int j, a[10],  * p2, *q2;

p = &x;
int *r = &rate;
```


Declare and initialize pointer

```
int *ptr;  /* declare a pointer to int  */
```

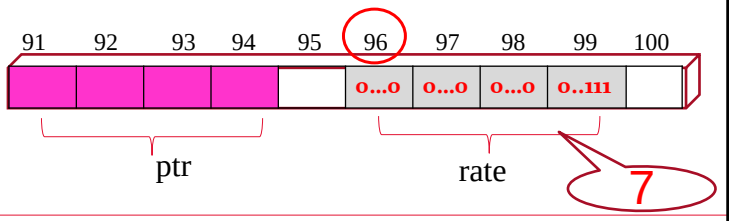
- Create a variable holding the address of other variable



Declare and initialize pointer

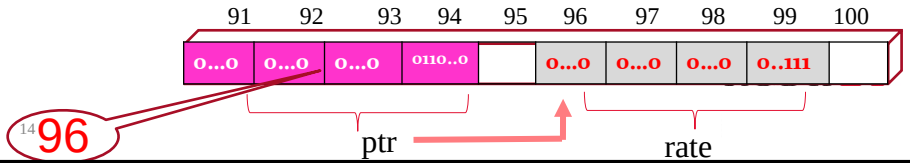
```
int *ptr;  /* declare a pointer to int  */
```

- Create a variable holding the address of other variable




```
ptr = &rate;  /*assigning address of rate*/
```

- Store address/pointer of **`rate`** in **`ptr`** (i.e., **`ptr`**'s value is the address)
- **`ptr`** now 'points to' **`rate`**



14

```
int *ptr;     /* I'm a pointer to an int */
```


```
ptr = &rate; /*I got the address of rate */
```

15

```
int *ptr;      /* I'm a pointer to an int */
```


```
ptr = &rate; /*I got the address of rate */
```


```
*ptr;      /* dereferencing. Indirect access.
              Get content (value) of the pointee */
```

`ptr`	`&rate`	address of `rate`
`*ptr`	`rate`	content (value) of `rate`

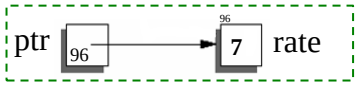
"mnemonic"

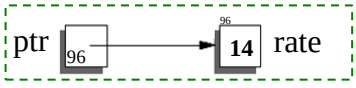
```
printf("%d", rate);   // 7  "direct access"
printf("%d", *ptr);   //  7  "indirect access"
```

16

16

```
int main()
{
  int rate = 7;
  int *ptr = &rate;
  printf("%d\n", rate); /* 7 */
  printf("%d\n", *ptr);

  int i = *ptr; // i=rate

  *ptr =  14; // rate = 14

  printf("%d %d\n", rate, *ptr);

  printf("%p %p\n", &rate, ptr);
}
```

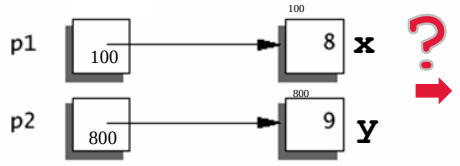




17

Some example of Pointer basics

```
int *p1, *p2;  int x = 8, y = 9;
p1 = &x;  p2 = &y;
*p1 = *p2;    // x = y
```



Assume x is at address 100, y is at address 800

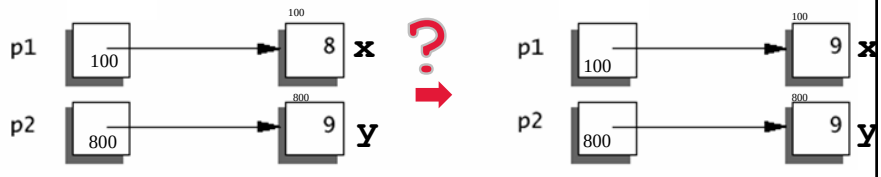
```
// copy value of p2's pointee (y) into pointee of p1 (x)
   *p1 is the alias of x    *p2 is the alias of y
```

18

18

Some example of Pointer basics

```
int *p1, *p2;  int x = 8, y = 9;
p1 = &x;  p2 = &y;
*p1 = *p2;    // x = y
```



Assume x is at address 100, y is at address 800

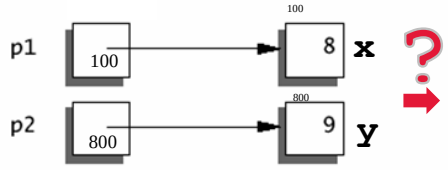
```
// copy value of p2's pointee (y) into pointee of p1 (x)
   *p1 is the alias of x    *p2 is the alias of y
```

19

19

Some example of Pointer basics

```
int *p1, *p2;  int x = 8, y = 9;
p1 = &x;  p2 = &y;
p1 = p2;   /*copy the content of p2 (address of y) into p1
             now p1 also points to y  */
```



Assume x is at address 100, y is at address 800

Java:
```
Student s1 = new Student("John", 22);
Student s2 = new Student("Gorge",20);
s1 = s2;
```

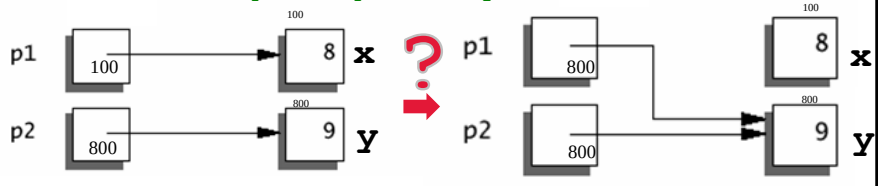
RECALL

20

20

Some example of Pointer basics

```
int *p1, *p2;  int x = 8, y = 9;
p1 = &x;  p2 = &y;
p1 = p2;   /*copy the content of p2 (address of y) into p1
             now p1 also points to y  */
```



Assume x is at address 100, y is at address 800

Java:
```
Student s1 = new Student("John", 22);
Student s2 = new Student("Gorge",20);
s1 = s2;
```

RECALL

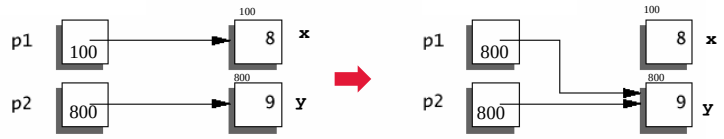
21

21

Some example of Pointers -- summary

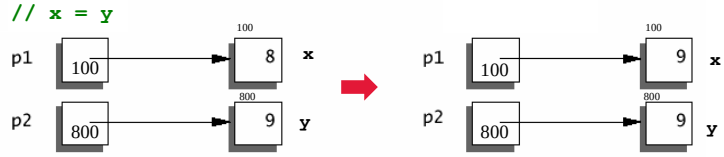
```
int *p1, *p2, x = 8, y = 9;
p1 = &x;  p2 = &y;
```

```
p1 = p2; // p1 = &y
```




```
printf("%d %d\n", *p1, *p2); // 9 9
printf("%p %p\n",  p1, p2); //  800 800
```

```
*p1 = *p2; // x = y
```




```
printf("%d %d\n", *p1, *p2); // 9 9
printf("%p %p\n",  p1, p2); //  100 800
```

22

22

- Lab4 posted
  - Academic honesty
- SMQ1 tonight
  - individual work discussion not allowed
- Assignment1 soon
  - individual work discussion not allowed

23

23