

# POINTERS IN C, PASSING POINTERS TO FUNCTIONS

CSSE 120—Rose Hulman Institute of Technology

# Can functions modify actual parameters?

- In Python, no! (pass by value)

- Consider this function:

```
void downAndUp(int takeMeHigher, int putMeDown) {  
    takeMeHigher += 1;  
    putMeDown -= 1;  
}
```

- How is this C function invoked?

  - ▣ `downAndUp(up, down);`

- Will calling the function change the values of the actual parameters **up** and **down**?

# References to simple variables

- Pointers need to be used to change the value of an argument
- A **pointer** is a variable that holds the **address** of a variable

```
int num = 4;
```

```
int *pNum;
```

```
pNum = &num;
```

```
*pNum == num == 4
```

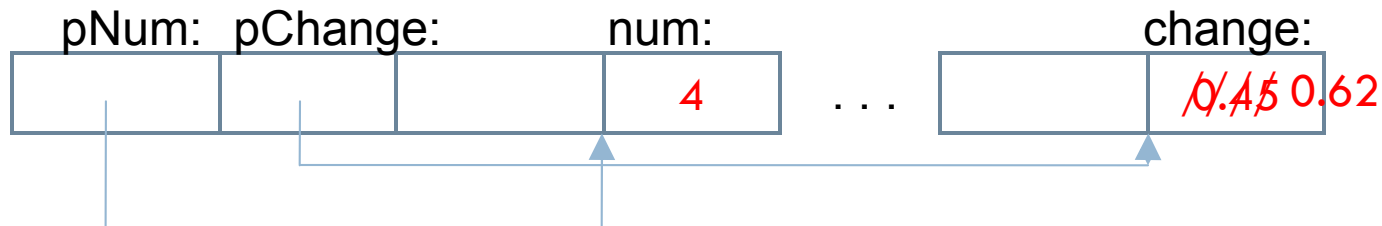
& is the address operator

```
double change = 0.45;
```

```
double *pChange;
```

```
pChange = &change;
```

```
*pChange = .62;
```



# Pass pointers to values to be changed

- How do we modify `downAndUp()` so that it actually changes the values of its parameters?
  - ▣ By passing pointers to the parameters to be changed
- Together, implement a function that passes pointers to values to be changed
  - ▣ Checkout project `PointersInclass` from SVN
  - ▣ Implement `downAndUpthatWorks()`
  - ▣ Use function `testdownAndUpthatWorks()` to test `downAndUpthatWorks()`

# Pointer Assignments

```
int x=3, y = 5;
int *px = &x;
int *py = &y;
printf("%d %d\n", x, y);
*px = 10;
printf("%d %d\n", x, y); /* x is changed */
px = py;
printf("%d %d\n", x, y); /* x not changed */
*px = 12;
printf("%d %d\n", x, y); /* y is changed */
```

# Break



- Starring **Binky!**
- (See <http://cslibrary.stanford.edu/104/>)

# Pointer Pitfalls

- Don't try to dereference an unassigned pointer:
  - ▣ `int *p;`  
`*p = 5;        /* oops! Program probably dies! */`
- Pointer variables must be assigned *address* values.
  - ▣ `int x = 3;`  
`int *p;`  
`p = x /* oops, RHS should be &x */`
- Be careful how you increment
  - ▣ `*p +=1;        /* is not the same as ... */`
  - ▣ `*p++;`

# In-class exercise on pointer pitfalls



- The rest of today's quiz lets you see some pointer pitfalls in action. These make great exam questions!
- Do it now
  
- When you are done, start the homework:
  - ▣ More pointer output
  - ▣ Writing functions to change variables
    - doubleMe
    - swap
  - ▣ scanf revisited