CS 161
Intro to CS I

Continue Functions:
Pass by Value, Reference, Pointer
Odds and Ends...

- Design due Sunday at 5pm on Canvas.
- EECS Networking night tonight, 5-8pm (KEC)
- Career Fair Thursday 11am-3pm, CH2M Hill Alumni Center
C++ Pass by Value

```cpp
void swap(int, int);

int main() {
    int a=5, b=10;
    swap(a, b);
    cout << "a:" << a << "b:" << b;
}

void swap(int x, int y) {
    int temp = x;
    x = y;
    y = temp;
}
```

- What if we didn’t have temp?
Variables vs. Pointers

- **Value Semantics**
  - Values stored directly
  - Copy of value is passed
  ```
  int i, j=2;
  i=j;
  ```

- **Pointer Semantics**
  - Address to variable is stored
  - Copy of address is passed
  ```
  int *i, j=2;
  i=&j;
  ```
void swap(int *, int *);

int main() {
    int a=5, b=10;
    swap(&a, &b);
    cout << "a: " << a << " b: " << b;
}

void swap(int *x, int *y) {
    int temp = *x;
    *x = *y;
    *y = temp;
}
C++ Pass by Reference

```cpp
void swap(int &, int &);
int main() {
    int a=5, b=10;
    swap(a, b);
    cout << "a: " << a << "b: " << b;
}
void swap(int &x, int &y) {
    int temp = x;
    x = y;
    y = temp;
}
```

Pointer and References Cheat Sheet

• *
  – If used in a declaration (which includes function parameters), it creates the pointer.
    • Ex. int *p; //p will hold an address to where an int is stored
  – If used outside a declaration, it dereferences the pointer
    • Ex. *p = 3; //goes to the address stored in p and stores a value
    • Ex. cout << *p; //goes to the address stored in p and fetches the value

• &
  – If used in a declaration (which includes function parameters), it creates and initializes the reference.
    • Ex. void fun(int &p); //p will refer to an argument that is an int by implicitly using *p (dereference) for p
    • Ex. int &p=a; //p will refer to an int, a, by implicitly using *p for p
  – If used outside a declaration, it means “address of”
    • Ex. p=&a; //fetches the address of a (only used as rvalue!!!) and store the address in p.
In-class Exercise

Understanding Pointers

• Create a pointer to a double, i.e. `double *d;` and three doubles `d1`, `d2`, and `d3` that get the values 7.8, 10.0, and .009.

• Now, set the pointer, `d`, to point to each double variable, `d1`, `d2`, and `d3`, printing the address and contents of each double variable along the way.
Demo...
In-class Exercise

Understanding Pointers

• What if you made a pointer that points to a pointer to a double, i.e. double **dp? Now, set dp to point to d, and use dp to print the address and contents of each double variable!!!
Demo...