Class 15

Reference Parameters

Void function

```
int positiveCube(int a){
  if(a < 0) return a * a * a * -1;
  else return a * a * a;
}

int main(){
  int a, b;
  cout << “Enter two numbers: “;
  cin >> a >> b;
  // update each to store the positive cube
  a = positiveCube(a);
  b = positiveCube(b);
  cout << a << “ “ << b << endl;
  return 0;
}
```

```
void positiveCubes(int &a, int &b){
  if(a < 0) a = a * a * a * -1
  else a = a * a * a;
  if(b < 0) b = b * b * b * -1;
  else b = b * b * b;
}

int main(){
  int a, b;
  cout << “Enter two numbers: “;
  cin >> a >> b;
  // update each to store the positive cube
  positiveCubes(a, b);
  cout << a << “ “ << b << endl;
  return 0;
}
```

Call by value

- When passing values to a function, C++ creates a copy of the values stored in the variable
- The function operates on those copies of values

Call by reference

- When you want to pass the actual variable to the function, you mark this in the title line by putting an & between the type and name of the parameter

Example for discussion

```
// what happens when the parameter of printAddress is changed from int &a to int a?

void printAddress(int &a){
  cout << “a in printAddress contains " << a << endl;
  cout << "Memory location of a in printAddress is " << &a << endl;
}

int main()
  int x = 5;
  cout << “x in main contains " << x << endl;
  cout << "Memory location of x in main is " << &x << endl;
  printAddress(x);
  return 0;
}
```

Example 1

```
void applyCurve(int &score){
  score = score + 10;
}
int main(){
  int grade = 75;
  applyCurve(grade);
  cout << grade << endl;
  return 0;
}
```

Example 2

```
// Find the bug in the code

void doubleNum(int &a){
  cout << “Your number doubled is “ << 2 * a << endl;
}
int main(){
  cout << “My number is 15” << endl;
  doubleNum(15);
  return 0;
}
```

Example 3

```
void secretName(string  &name){
  int coinToss = rand()%2;
  if(coinToss == 0) name = “Freddy”;
  else name = “Bakugo”;
}

int main(){
  srand(time(0));
  string name;
  cout << “What is your name? “;
  cin >> name;
  secretName(name);
  cout << “Your name is actually “ << name << endl;
  return 0;
}
```

Example 4

```
void swap(int &a, int &b){
   int temp = a;
   a = b;
   b = temp;
}
void sortVarValues(int &a, int &b, int &c){
   if(a > c) swap(a, c);
   if(a > b) swap(a, b);
   if(b > c) swap(b, c);

}

int main(){
   int x = 7, y = 5, z = 10;
   sortVarValues(x, y, z);
   cout << x << “ “ << y << “ “ << z << endl;
}
```

Key summary

- Call by value parameter:
  - A copy of the value is passed
  - Changes made to the value inside the function are not permanent
  - An argument can be a hard-coded number, for example:
    - sqrt(5.0);
- Call by reference parameter:
  - Changes are permanent
  - A call by reference argument must be a variable