Class 14

Reference Parameters

Re-cap: Writing Functions

- Need to include function definition before the main function
- Before you code any function, plan how it will be used in a simple main function
- Doing that clarifies the name, inputs, and result type needed

Examples

1. Find the max of two numbers
2. Determine if input is divisible by 15
3. Calculate the area of a rectangle
4. Calculate the area and perimeter of a rectangle

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

Example 1

```
int product(int a, int b){
   a = a * b;
   return a;
}

int main(){
   int x = 5, y = 6;
   cout << product(x, y) << endl;
   return 0;
}
```

Example 2

```
void swap(int a, int b){
   int temp = a;
   a = b;
   b = temp;
}
int main(){
   int x = 5, y = 3;
   swap(x, y);
   cout << “x = “ << x << “; y = “ << y << endl;
   return 0;
}
```

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 3

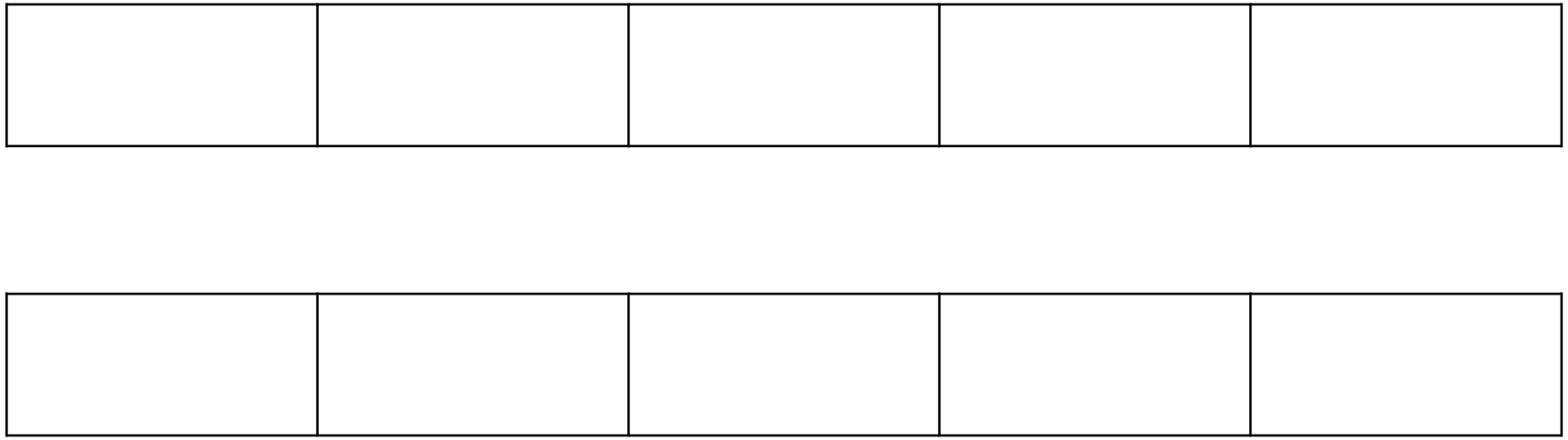
```
void swap(int &a, int &b){
   int temp = a;
   a = b;
   b = temp;
}
int main(){
   int x = 5, y = 3;
   swap(x, y);
   cout << “x = “ << x << “; y = “ << y << endl;
   return 0;
}
```

Call by value v. call by reference

```
int main(){
   int x = 5, y = 3;
   swap(x, y);
   cout << ”x = “ << x << endl;
   cout << “y = “ << y << endl;
   return 0;
}
```

```
void swap(int a, int b){
   int temp = a;
   a = b;
   b = temp;
}
```

```
void swap(int &a, int &b){
   int temp = a;
   a = b;
   b = temp;
}
```



Example 4

```
// 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 5

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

Example 6

```
void secretName(string  &name){
   int coinToss = rand()%2;
   if(coinToss == 0) name = "Bob";
   else name = "Sandy";
}
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 7

```
// 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 8

```
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;
   return 0;
}
```

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