Parameters: pass-by-value, -reference

- our “usual” parameter passing mechanism is called pass-by-value
- it evaluates the value being passed, and copies that value to the function's parameter

```
void displayVal(double val)
{
    cout << “Result is”;
    cout << val;
    cout << endl;
}
```

```
int main()
{
    double x = 100;
    double y = 4.5;
    displayVal(x);              // copies 100 to parameter val
    displayVal(y);              // copies 4.5 to parameter val
    displayVal(x * y + 10); // copies 114.5 to parameter val
}
```

Functions alter -by-value params

- a called function can assign a value to it's pass-by-value parameter, but it's just altering the copy

```
void changeA(int a)
{
    a = 23; // change param
}

int main()
{
    int x = 10;
    changeA(x);
    cout << x; // still 10
}
```

Pass-by-reference

- we can add extra syntax (an &) in the definition of a parameter to make it pass-by-reference
- means the function can actually alter the value of variables passed to it

```
void changeA(int &a)
{
    a = 23; // alters actual passed param
}
int main()
{
    int x;
    changeA(x);
    cout << x; // x is now 23
}
```

Pass-by-ref continued

- can only pass a variable to a pass-by-ref parameter, and it must be of the same type

```
void changeA(float &a)
{
    cout << “Enter new number”;
    cin >> a;
}

int main()
{
    int x;
    change(x); // fails, x and a are diff types
    change(10*x);  // fails, 10*x is not a variable
}
```

Pass-by-ref uses

- a function can only return one value through the return statement
- often we would like a function to get multiple values for us
- done by passing “by-ref” parameters for the others

```
// function to get x,y coords
void getCoords(int &x, int &y)
{
    cout << “Enter the x and y coordinates, “;
    cout << “e.g. 51 12”;
    cin >> x;
    cin >> y;
}
```

```
int main()
{
    int x1, y1, x2, y2;
    // get coordinates for two points
    getCoords(x1,y1);
    getCoords(x2,y2);
}
```

Avoid indiscriminate pass-by-ref

- If a parameter doesn't actually need to be pass-by-reference then don't use pass-by-ref
- this prevents functions from accidentally changing values that we don't want changed
- simplifies debugging/maintenance by eliminating some possible errors