FUNCTIONS AND THE CONST KEYWORD

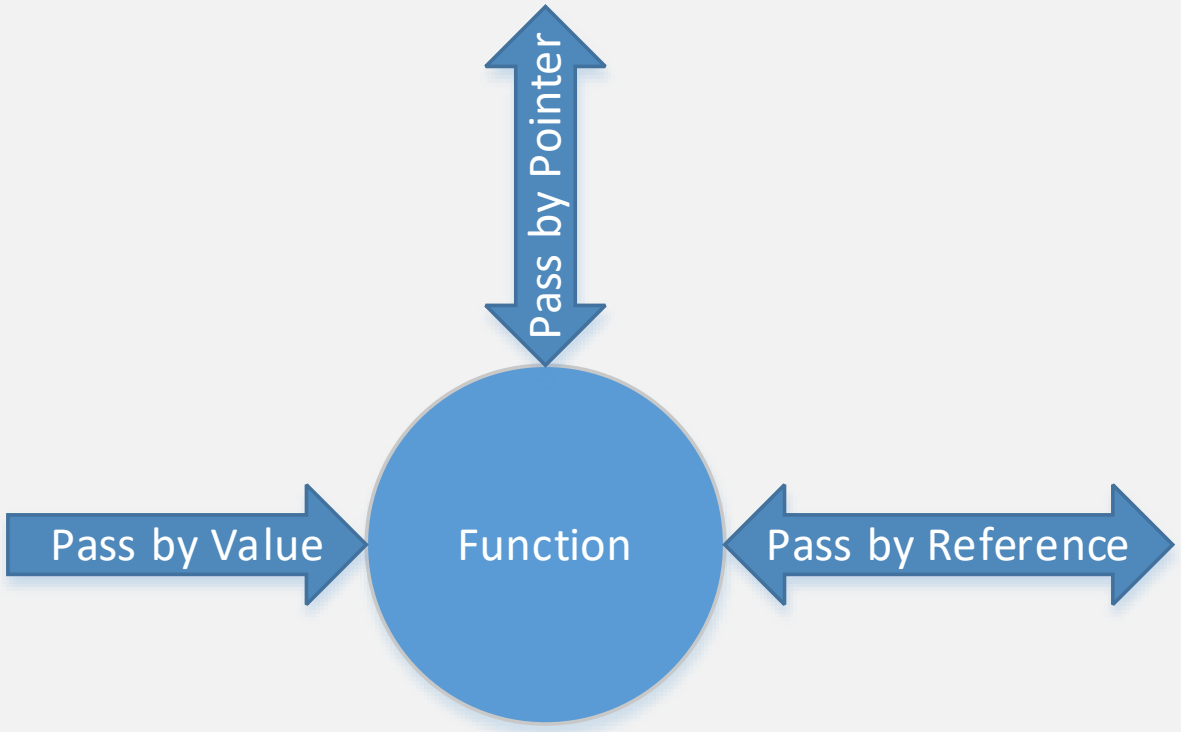
Preventing Data Change:

const Arguments

const Return Values

Delroy A. Brinkerhoff

INPUT ONLY VS. INOUT



PASS BY VALUE

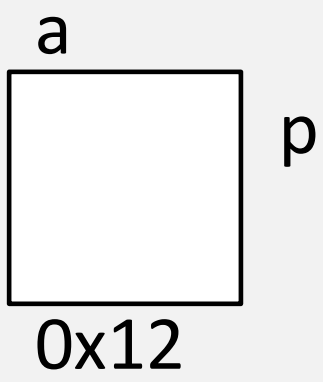
- Pass by value (aka pass by copy) only allows data input
- Input only is appropriate for some functions

```
table(payment, principal, R, N);
```

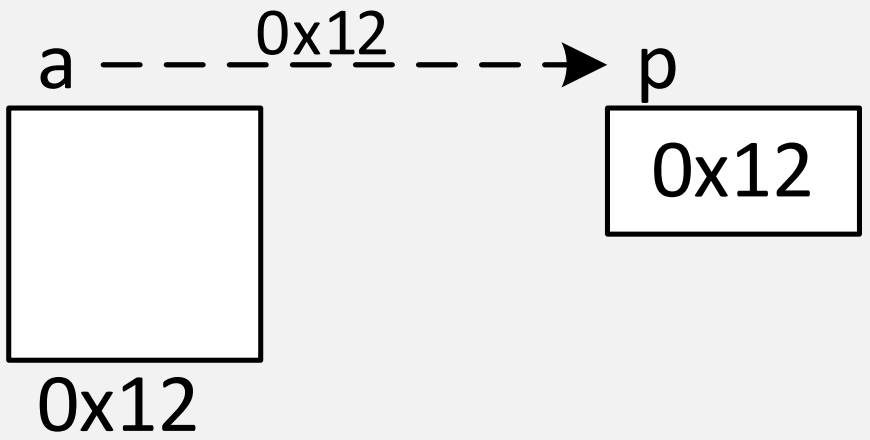
```
void table(double payment, double balance, double R, int N)
{
    for (int i = 1; i <= N; i++)
    {
        balance -= (payment - to_interest);
    }
}
```

FAST & EFFICIENT DATA PASSING

REFERENCE



POINTER



PASS BY REFERENCE

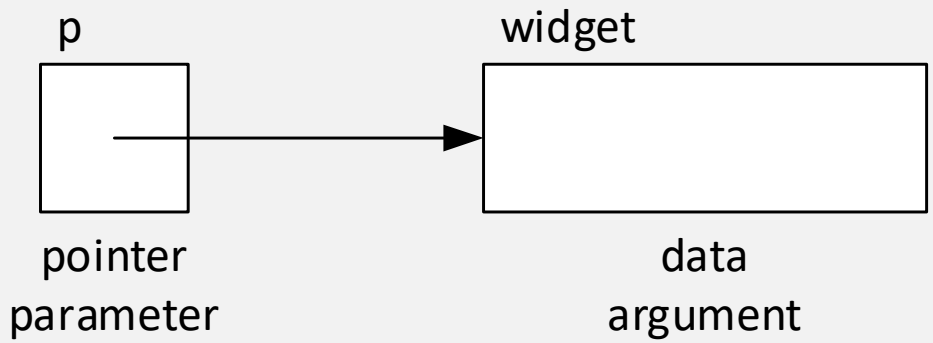
```
struct ReallyBig
{
    char   code;
    double cost;
        . . .
};
```

```
void function(const ReallyBig& big)
{
    double tax = big.cost * 0.077;
    //big.code = 'Z';
}
```

PASS BY POINTER

- `ReallyBig widget;`
- `function1(&widget);`
- `function2(&widget);`
- `function3(&widget);`

- `void function1(const ReallyBig* p);`
- `void function2(ReallyBig* const p);`
- `void function3(const ReallyBig* const p);`



RETURNING A CONST REFERENCE

```
const ReallyBig& function1()
{
        static ReallyBig widget =
                { 'x', 19.95, ... };
        return widget;
}
```

```
const ReallyBig& r1 = function1();

//r1.cost = 29.95;  // error
```

RETURNING CONSTANT DATA

```
const ReallyBig* function1()
{
        static ReallyBig widget =
                { 'x', 19.95, ... };
        return &widget;
}
```

```
const ReallyBig* p1 = function1();

//p1->cost = 29.95;          // error
p1 = new ReallyBig;          // okay
```

RETURNING A CONSTANT POINTER

```
ReallyBig* const function2()
{
        static ReallyBig widget =
                { 'x', 19.95, ... };
        return &widget;
}
```

```
ReallyBig* const p2 = function2();

p2->cost = 29.95;           // okay
//p2 = new ReallyBig;       // error
```