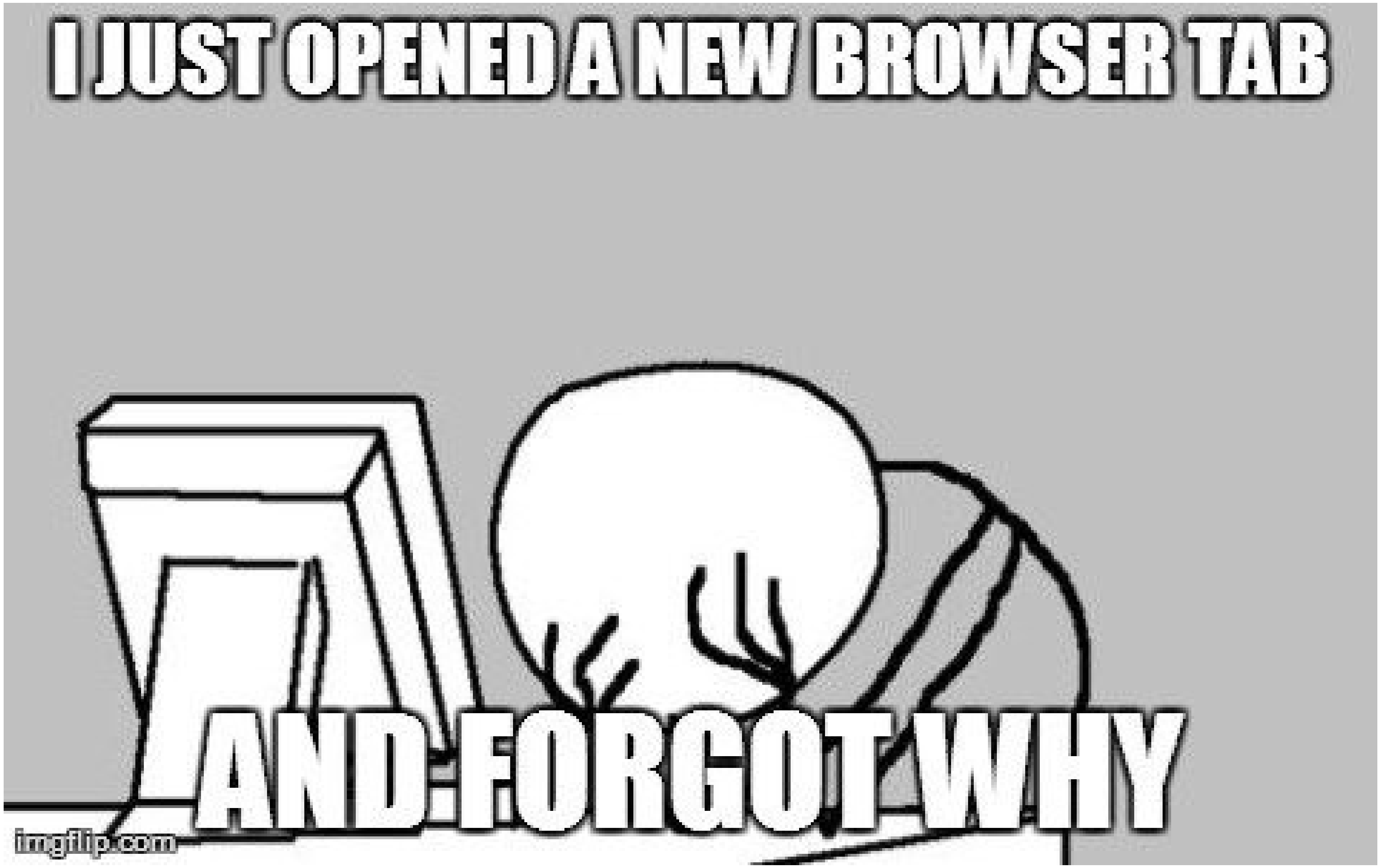
I JUST OPENED A NEW BROWSER TAB
AND FORGOT WHY
imgflip.com

Passing by Reference

CS2308
Gentry Atkinson

Passing by Value

- The arguments of a function call are copied into the memory allocated to a function parameters.
- The arguments and the parameters are different variables in different scopes.
- Changes to parameters do not change the arguments.

Example 1

```
void foo(int a){
    a = 2*a;
    cout << "Value of a in foo: "
        << a << endl;
}

int main(int argc, char** argv){
    int b = 2;
    cout << "Value b before foo: "
        << b << endl;
    foo(b);
    cout << "Value b after foo: "
        << b << endl;
} //try to predict the output
```

Memory

b: 2

0

0

0

0

0

0

0

a: ~~2~~ 4

0

0

0

0

0

0

Passing by Reference

- The <u>address</u> of an argument is passed to the parameter. They are both stored in the same memory location.
- The arguments and the parameters are still different variables in different scopes.
- Changes to the parameter <u>do</u> change the argument.

Example 2

```
void foo(int &a){
    a = 2*a;
    cout << "Value of a in foo: "
        << a << endl;
}

int main(int argc, char** argv){
    int b = 2;
    cout << "Value b before foo: "
        << b << endl;
    foo(b);
    cout << "Value b after foo: "
        << b << endl;
} //try to predict the output
```

Memory
0
0
0
0
0
b, a: ~~2~~ 4
0
0
0
0
0
0
0
0
0
0

• When to Pass by Reference

- **Large objects like arrays and structs should be passed by reference to improve efficiency.**
- **Arrays are passed by reference by default.**
- **Passing by reference can let a function use parameters as outputs.**
- **Passing by value is safer. const should be used to pass by reference safely, when possible.**

Example 3

```
void foo(const int a[], int &b, int &c, int SIZE){
    b =  c = a[0];
    for(int i = 1; i < SIZE; i++){
        if(a[i] < b) b = a[i];
        if(a[i] > c) c = a[i];
    }
}

int main(int argc, char** argv){
    int a[] = {2, 6, 4, 8, 3, 4};
    int min, max;
    foo(a, min, max, 6);
    cout << "The minimum of a is " << min << endl;
    cout << "The maximum of a is " << max << endl;
} //try to predict the output
```

Example 4

```
struct Dog{
    string breed; string name; int age;
};

bool bigDog(const Dog &d){
    if(d.breed == "mastif"||
d.breed=="dane"){
        return true;
    }
    return false;
}
```

```
int main(int argc, char** argv){
    Dog d = {"mastif", "Jimmy", 6};
    if(bigDog(d))
        cout << d.name << " is a big
dog." << endl;
    else
        cout << d.name << " is a
small dog." << endl;
} //try to predict the output
```

Remember that arrays are <u>always</u> passed by reference!

Questions or Comments?