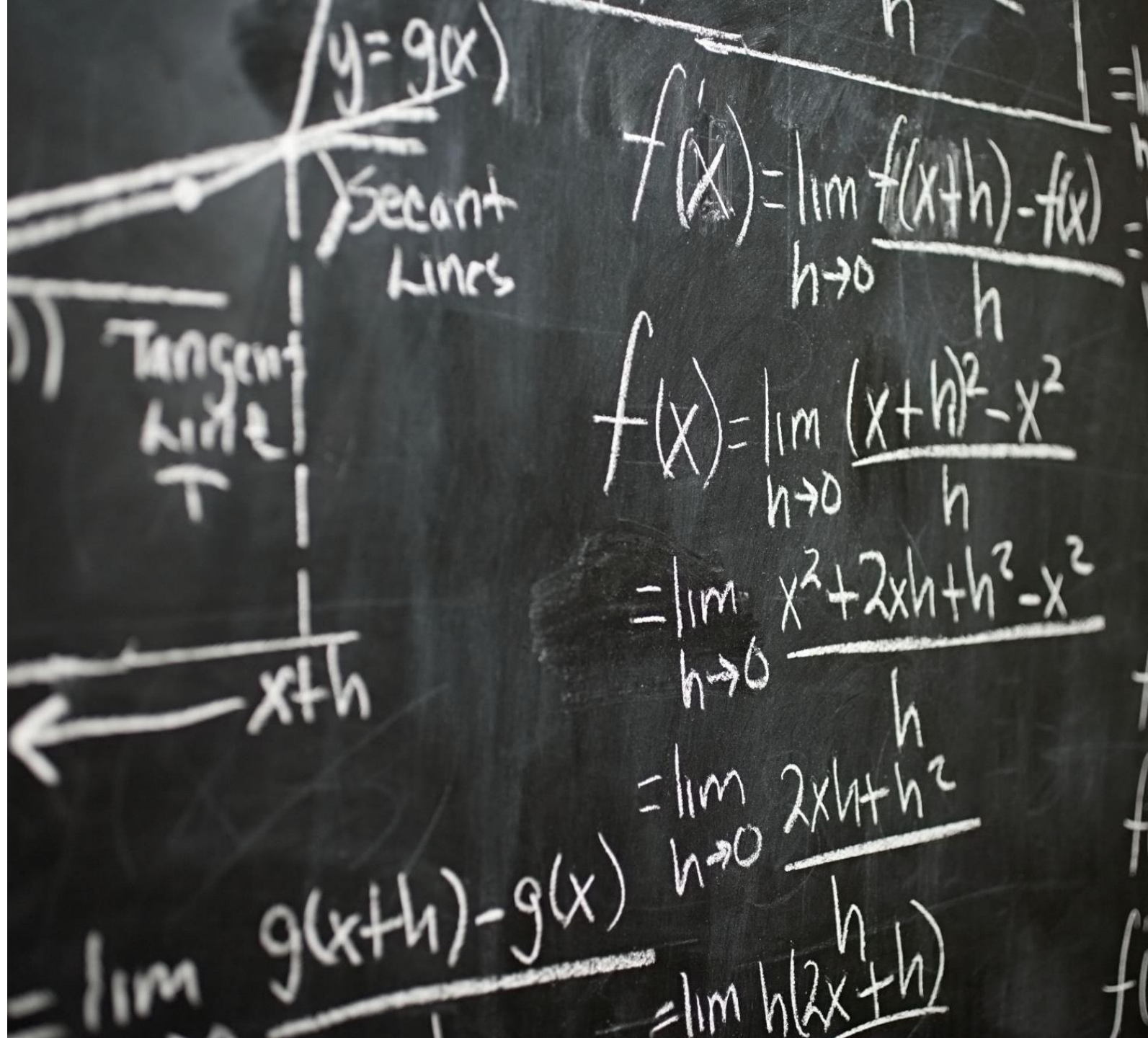


# FUNCTIONS

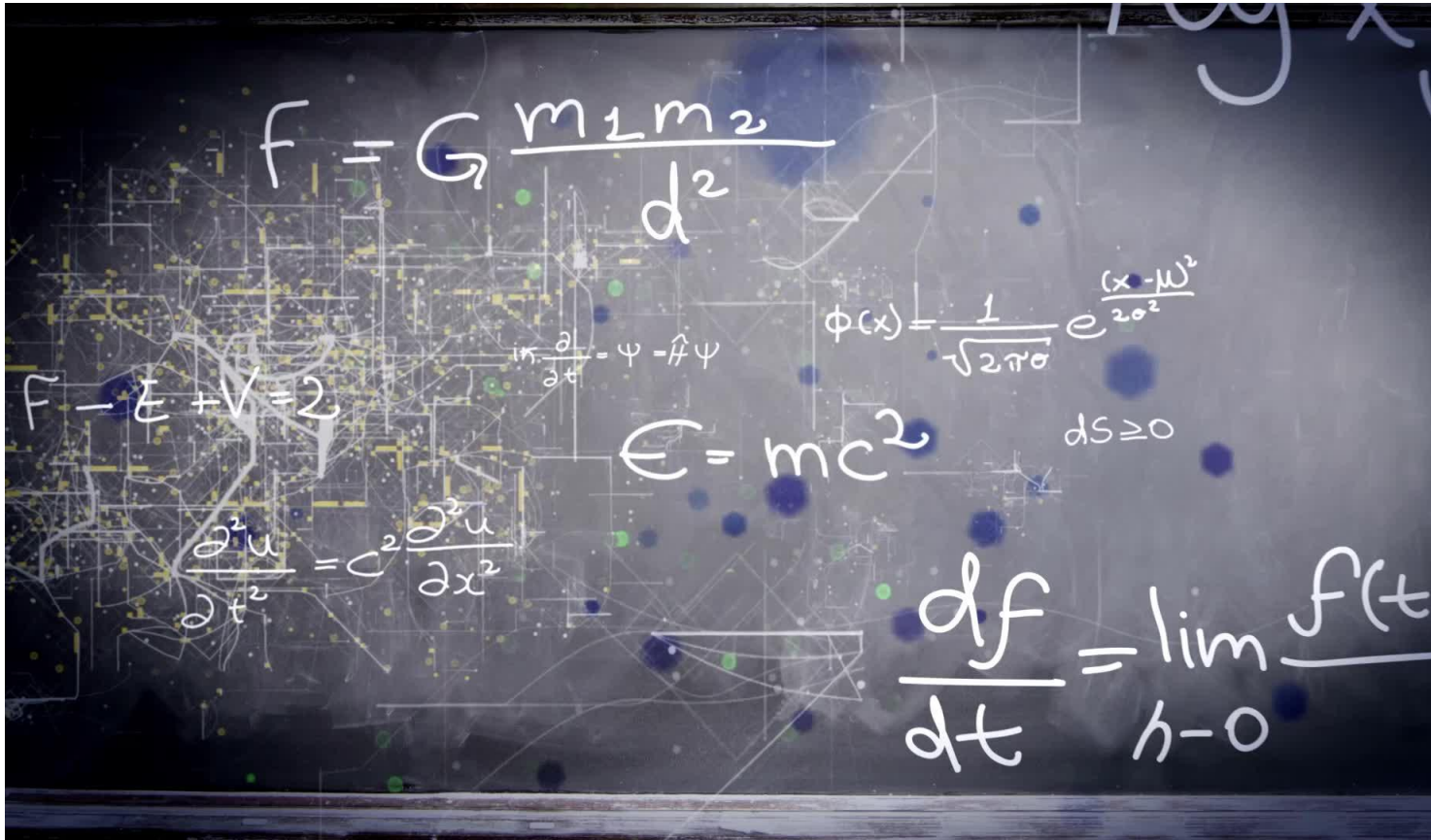
Prof. Andrew Sheehan

Boston University/MET  
Computer Science Dept.



# FUNCTIONS ARE OBJECTS

```
function Name of the object validateScreenName(Parametersname) {  
    /* This is the function body */  
}
```



# FUNCTION DECLARATIONS

```
function eqOfLine(m, x, b) {  
    return (m * x) + b;  
}
```



# RETURNING A VALUE

Implicitly  
returns  
undefined  
if you do  
not  
indicate a  
return  
value.


}

# NO SEMICOLON

```
function validateName(name) {
```

```
}
```

← Semicolon at the end of a function declaration is **not required**.

A yellow pushpin is pinned to a calendar page. The calendar shows dates like 29, 30, and 31. The background is a blurred blue and white.

# The arguments object

`arguments` is an `Array`-like object accessible inside [functions](#) that contains the values of the arguments passed to that function.

PARAMETERS AND  
FUNCTION ARGUMENTS |



```
const now = new Date();  
healthCheck(message, now);  
healthCheck(message);
```

```
function healthCheck(message) {  
  const timestamp =  
    arguments[1] || new Date();  
}
```

# PARAMETERS AND FUNCTION ARGUMENTS



```
function check(time) {  
    let timestamp = time || new Date();  
}
```

```
function health (message) {  
    message = message || 'none';  
}
```

```
function ping(health, check) {  
    heath("Pinging the server");  
    check();  
}
```

## PASSING FUNCTIONS



# NESTED FUNCTIONS

It is normal to declare a function within a function ('closure')

```
function greet(first, last) {  
  function getFullName() {  
    return first + " " + last;  
  }  
  alert("Hello, " + getFullName());  
  alert("Bye, " + getFullName());  
}
```



# SHADOWING

```
flag = `0n`; // Global
```

```
function isLucky() {  
    var flag = `0ff`; // Shadowed  
    return flag === `0n`;  
}
```

## OBJECT LITERALS AS NAMED PARAMETERS

```
entries({ start: 0, end: -1; step:0 });
```

```
function entries(options) {  
  options = options || {};  
  const start = options.start || 0;  
  const end = options.end || -1;  
  const step = options.step || 1;  
}
```

```
function sayHello(user = "Guest") {  
    return `<div>${user}</div>`;  
}
```

DEFAULT PARAMETERS



```
function adder (...values) {  
  let sum = 0;  
  values.forEach(value => {  
    sum += value;  
  })  
  return sum;  
}
```

---

USING **REST**  
PARAMETERS

# CALL STACK

Most recent function invocation will be on the top of the stack.

