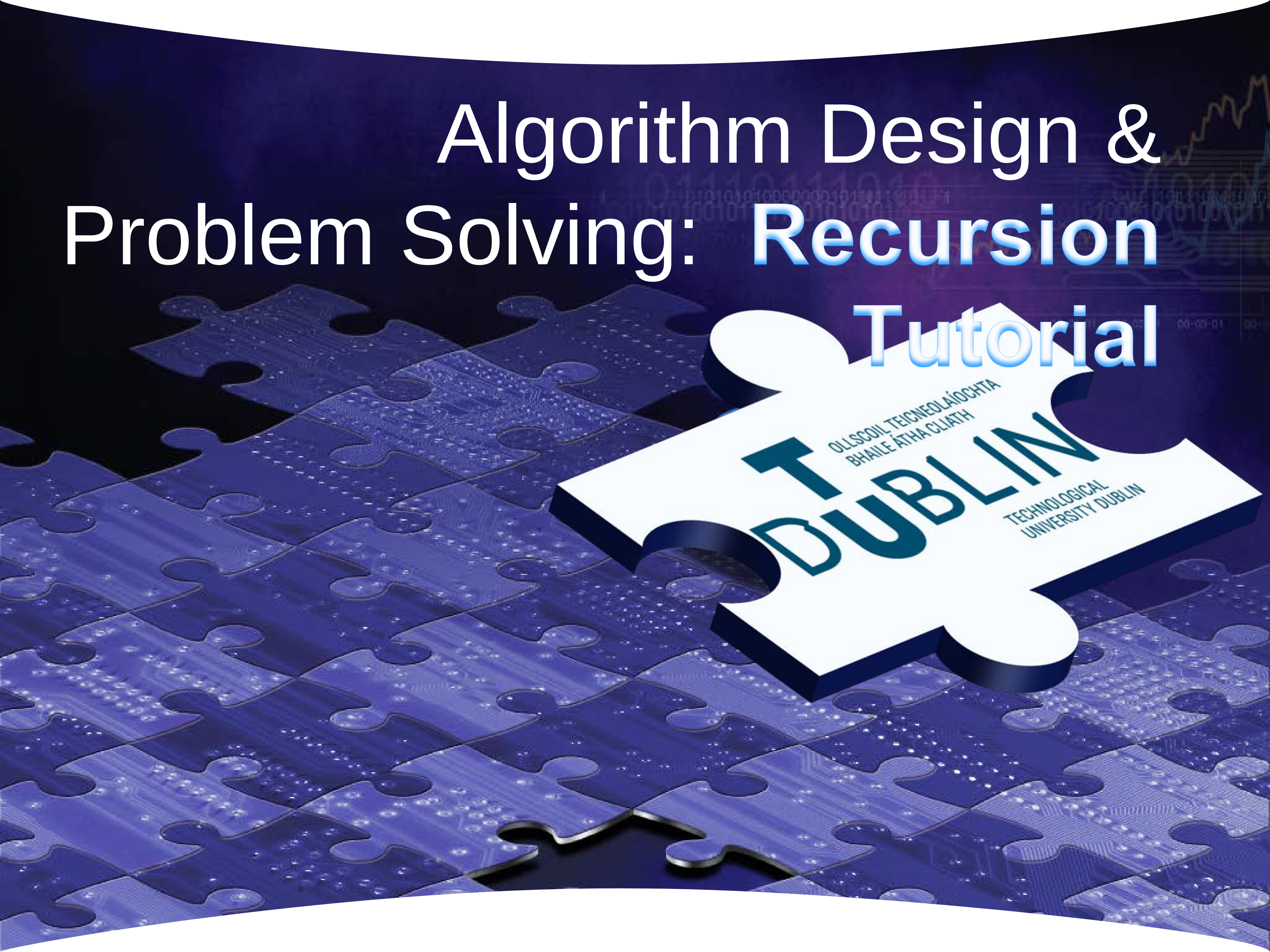
Algorithm Design & Problem Solving: **Recursion Tutorial**



When should I use Recursion?

❖ If the algorithm has a **base case**

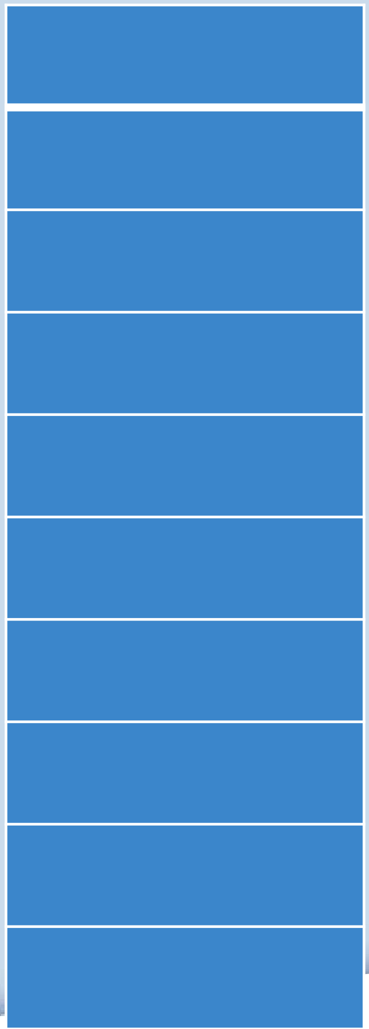
❖ If a problem is **iterative**

❖ If the problem gets **progressively smaller**

Recursive Factorial

Factorial 6 ***Call Stack***

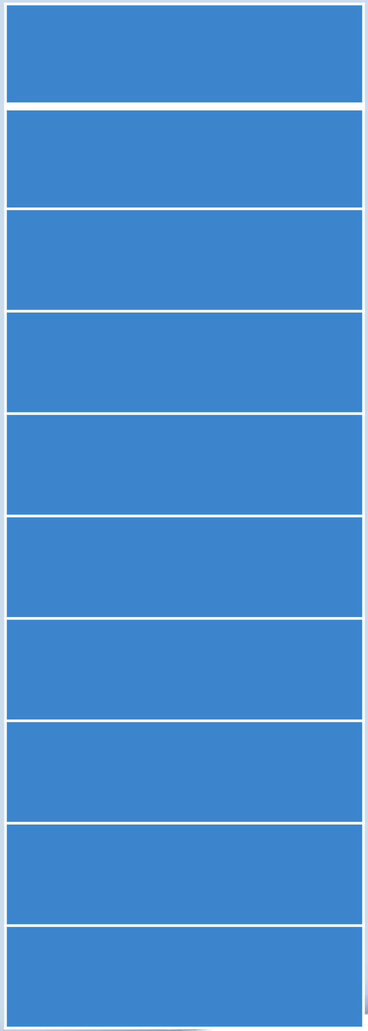
Algorithm



Recursive Power

Power (3, 6) ***Call Stack***

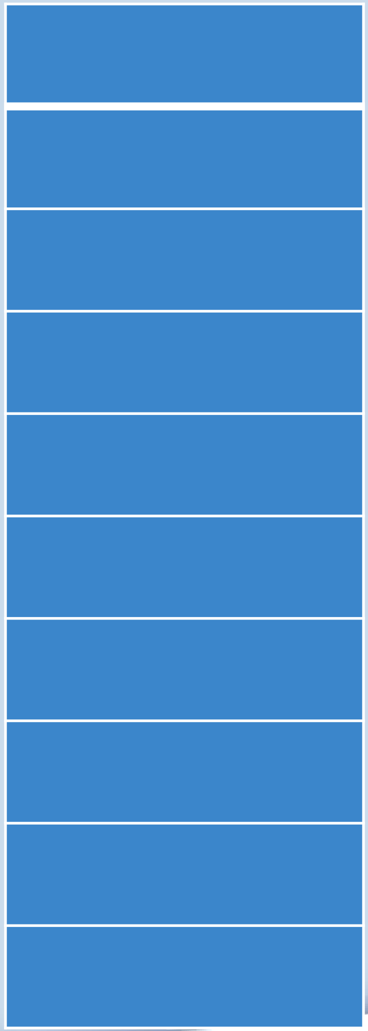
Algorithm



Recursive GCD

gcd (72,16) ***Call Stack***

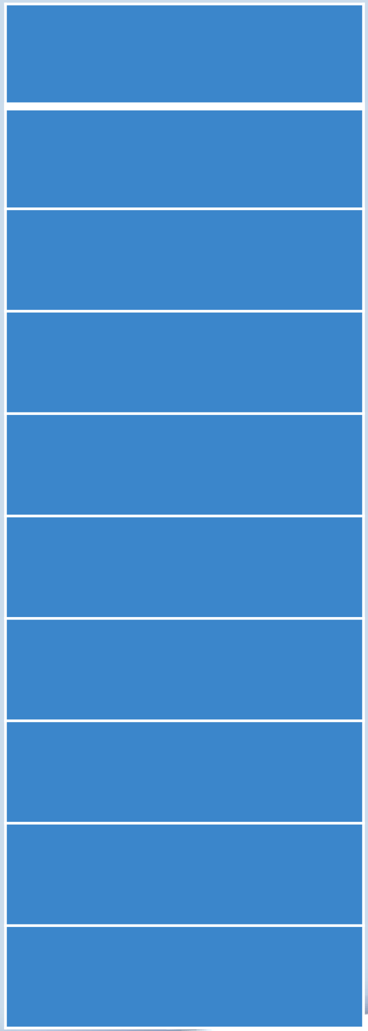
Algorithm



Recursive Fibonacci

fibonacci (4) ***Call Stack***

Algorithm



Reminder: Recursive Factorial

- **Remember this …. 4!**

```
Factorial (n)
    if n=1 or n=0
            return 1
    else
            return n*Factorial(n-1)
  end
```

Call Stack

BASE CASE REACHED - 1
2*Factorial(1)
3*Factorial(2)
4*Factorial(3)
Factorial(4)

Recursive Power Algorithm

```
begin power (x,y)
  if (y<1)
        return 1
  end if
  return x*power (x,y-1)
end
```

Recursive Euclid’s Algorithm

```
gcd(a, b)  //greatest common divisor or Euclid’s algorithm
 if (b = 0) then
    return a
else
    return gcd(b, a mod b)

end
```

Recursive Fibonacci

```
fibonacci(n)
if (n=0 or n=1)
    return n
else
    return fibonacci(n-1) + fibonacci(n-2)

end
```

Thank You !

OLLSCOIL TEICNEOLAÍOCHTA
BHAILE ÁTHA CLIATH
TU DUBLIN
TECHNOLOGICAL
UNIVERSITY DUBLIN