MACQUARIE University

*Department of Computing*

**ITEC625 Fundamentals of Computer Science**
**Workshop - Workshop - Time Complexity**

## Learning outcomes

By the end of this session, you will have learnt the basics about time complexity.

## Questions

1. What are the degrees of the following polynomials?

$$5x^3 + 3x - 7 \tag{1}$$

ANSWER: 3

$$5x - 2x^6 \tag{2}$$

ANSWER: 6

2. What are the time complexities of the following codes?

```
int x = 1;
if(x % 2 == 0) {
        x++;
}
else {
        x--;
}
```

ANSWER: $O(1)$

```
for(int i=0; i < 100; i++) {
        System.out.print(i+ "␣");
}
```

ANSWER: $O(1)$

```
for(int i=0; i < n; i++) {
        System.out.print(i+ "␣");
}
```

ANSWER: $O(n)$

```
for(int i=0; i < n; i+=2) {
        if(i%3 == 0) {
                System.out.print(i+ ".␣");
        }
        else {
                if(i%5 == 0) {
                        System.out.print(i+ "!␣");
                }
                else {
                        System.out.print(i+ "?␣");
                }
        }
}
```

ANSWER: $O(n)$

```
for(int i=1; i < n; i*=2) {
        System.out.print(i+ "␣");
}
```

ANSWER: $O(log_2(n))$

```
for(int i=0; i < n; i+=n/5) {
        System.out.print(i+ "␣");
}
```

ANSWER: $O(1)$

```
for(int i=6; i < n/2; i++) {
        System.out.print(i+ "␣");
}
```

ANSWER: $O(n)$

```
for(int i=n/3; i < n/2; i+=4) {
        System.out.print(i+ "␣");
}
```

ANSWER: $O(n)$

```
for(double i=1; i*i <=n; i++) {
        System.out.print(i+ "␣");
}
```

ANSWER: $O(\sqrt{n})$

```
for(int i=0; i < n; i++) {
        for(int k=0; k < n; k++) {
```

```
                System.out.println(i+"␣");
        }
}
```

ANSWER: $O(n^2)$

```
for(int i=0; i < n; i++) {
        for(int k=0; k < n; k++) {
                if(i%3 == 0) {
                        System.out.print(i+ ".␣");
                }
                else {
                        if(i%5 == 0) {
                                System.out.print(i+ "!␣");
                        }
                        else {
                                System.out.print(i+ "?␣");
                        }
                }
        }
}
```

ANSWER: $O(n^2)$

```
for(int i=n; i > 0; i-=2) {
        for(int k=1; k <= n; k+=3) {
                if(i%3 == 0) {
                        System.out.print(i+ ".␣");
                }
                else {
                        if(i%5 == 0) {
                                System.out.print(i+ "!␣");
                        }
                        else {
                                System.out.print(i+ "?␣");
                        }
                }
        }
}
```

ANSWER: $O(n^2)$

```
for(int i=0; i < n; i++) {
        for(int k=1; k <= n; k*=2) {
                if(i%3 == 0) {
                        System.out.print(i+ ".␣");
                }
                else {
                        if(i%5 == 0) {
                                System.out.print(i+ "!␣");
```

```
                        }
                        else {
                                System.out.print(i+ "?␣");
                        }
                }
        }
}
```

ANSWER: $O(n \times log_2(n))$

```
for(int i=1; i < n/2; i*=2) {
        for(int k=1; k < n/2; k*=2) {
                if(i%3 == 0) {
                        System.out.print(i+ ".␣");
                }
                else {
                        if(i%5 == 0) {
                                System.out.print(i+ "!␣");
                        }
                        else {
                                System.out.print(i+ "?␣");
                        }
                }
        }
}
```

ANSWER: $O((log_2(n))^2)$

```
for(double i=1; i<=n; i+=1.0/n) {
        System.out.print(i+ "␣");
}
```

ANSWER: $O(n^2)$

3. What is the time complexities for the method `foo` in each of the following codes?

```
void foo(int n) {
        for(int i=0; i < n; i++) {
                System.out.print(i+ "␣");
        }
}
```

ANSWER: $O(n)$

```
int foo(int n) {
        int result = 0;
        for(int i=0; i < n; i++) {
                result+=bar(n);
        }
        return result;
}
```

```
int bar(int n) {
        return n%2;
}
```

ANSWER: $O(n)$

```
int foo(int n) {
        int result = 0;
        for(int i=n; i > 0; i--) {
                result+=bar(i);
        }
        return result;
}

int bar(int n) {
        return n%2;
}
```

ANSWER: $O(n)$

```
int foo(int n) {
        int result = 0;
        for(int i=n; i > 0; i--) {
                result+=bar(i);
        }
        return result;
}

int bar(int n) {
        int total = 0;
        for(int i=1; i<=n; i+=2) {
                total+=i;
        }
        return total;
}
```

ANSWER: $O(n^2)$

```
int foo(int[] arr) {
        for(int i=1; i < arr.length; i++) {
                if(arr[i] < arr[i-1]) {
                        return false;
                }
        }
        return true;
}
```

ANSWER: Best case: $O(1)$ (when first item is less than second item) Worst case: $O(n)$ (when each item is more than or equal to the item after it)

4. Write a piece of code with $O(log_2 n)$ time complexity.

   ANSWER:

```
int total = 0;
for(int i=1; i<=n; i*=2) {
        total = total + i;
}
```

5. Write a piece of code with a best case time complexity of $O(n)$ and worst case time complexity of $O(n^2)$.

   ANSWER:

```
int foo(int[] a) {
        for(int i=0; i < a.length; i++) {
                if(a[i]%2 == 0) {
                        for(int k=0; k < a.length; k++) {
                                result+=a[k];
                        }
                }
        }
        return result;
}
```

   Best case $O(n)$ when all numbers are odd Worst case $O(n^2)$ when all numbers are even

6. Is it possible to have a code with a best case time complexity of $O(n^2)$ and worst case time complexity of $O(n * log_2 n)$?

   ANSWER: No, best case time complexity cannot be worse than worst case time complexity.