# ASSIGNMENT 1

1. 3 Dimensions of Variability Across Programming Paradigms**:**

   - **<u>Control Flow:</u>** how execution flows within the program (sequence and branches, in concurrent threads, in reactive manner, declarative).
   - **<u>Code Organization:</u>** how code is organized into a hierarchy of units (expressions, functions, modules, packages) and how these units are organized.
   - **<u>Performance:</u>** how code can be run fast, use less resources (RAM, disk, network), behave better (responsive, scalable) at runtime.

2. (a) (x: number | string, y: number | string) => (x + y): number | string

   (b) x: T [] => x [0]: T

   (c) (x: boolean, y: number) => (x? y : -y): number

3. "Shortcut Semantics" is a programming concept that describes code semantics that saves calculations when any code execution is satisfactory. It's a concept for write a function more efficiently without affect it's correctness/meaning. We can find that two functions are not equivalent, one is using "Shortcut Semantics" and the other is not, by throwing errors for example:

   The function "every" of JavaScript is using this concept so the output for the next code is **false** because the function will stop after will be founded one element that does not satisfy the predicate:

```
const isNonNegative = (x: number)=>{if (x<0) return false; else throw true;}
let arr = [-1, 1, 1]
try {
    arr.every(isNonNegative);
}
catch (e) {
    e;
}
```

   While the next implementation of "every" function is not using this concept, because it scanning all the array, so it will throw **true**:

```
const everyImpl = (pred, arr) => arr.map(pred).reduce((acc, cur) => acc && cur, true)
try {
    everyImpl (isNonNegative, arr);
}
catch (e) {
    e;
}
```