KEEP CALM & USE FUNCTIONS

SCRIPT DE COMMANDE

```
const face = (d:Direction) : Command => ({kind: "Face", direction: d})
const start = () : Command => ({kind: "Start"})
const stop_ = () : Command => ({kind: "Stop"})
const chain = (first: Command, second: Command) : Command => ({kind: "Chain", first, second})

const pipe = (first: Command, ...rest: Array<Command>) : Command =>
    rest.length > 0
    ? chain(first, pipe(rest[0], ...rest.slice(1)))
    : first
```

MIAGE M2 - QUALITÉ DU SI - THOMAS HAESSLÉ & QUENTIN BURG

```
const moveWestNorth = pipe(
    face("West"),
    start(),
    stop_(),
    face("North"),
    start(),
    stop_()
)
```

SCRIPT DE COMMANDE

KEEP CALM & USE FUNCTIONS 

```
const face = (d:Direction) : Command => ({kind: "Face", direction: d})
const start = () : Command => ({kind: "Start"})
const stop_ = () : Command => ({kind: "Stop"})
const chain = (first: Command, second: Command) : Command => ({kind: "Chain", first, second})

const pipe = (first: Command, ...rest: Array<Command>) : Command =>
    rest.length > 0
    ? chain(first, pipe(rest[0], ...rest.slice(1)))
    : first
```

```
const moveWestNorth = pipe(
    face("West"),
    start(),
    stop_(),
    face("North"),
    start(),
    stop_()
)
```

PIPE OPERATION

REMINDER

Beaucoup de langages utilisent l'opérateur infix |> pour la fonction pipe

```
pipe(x, foo) = x |> foo = foo(x)
```

Il existe une proposition TC39 en stage 2 pour JavaScript https://github.com/tc39/proposal-pipeline-operator

Stage 2 étant encore incertain, je préfère utiliser la fonction pipe en JS/TS, nous utiliserons l'implémentation de la librairie fp-ts. Mais vous pouvez utiliser |> avec un polyfill.