# 20CYS312 - Principles of Programming Languages - Lab Exercise

---

**Name**: R Subramanian
**Roll Number**: CH.EN.U4CYS22043
**Date**: 29/11/2024

## Lab 1: Introduction to Programming Paradigms

### 1. Basic Arithmetic

**Objective**: Get familiar with GHCi and basic arithmetic operations.

#### Exercise 1:

Open GHCi and perform basic arithmetic operations:

```
3 + 5
10 * 4
6 / 2
```

**Output**:

```
subramanian@arch:~$ ghci
GHCi, version 8.8.4: https://www.haskell.org/ghc/  :? for help
Prelude> 5+3
8
Prelude> 10*4
40
Prelude> 6/2
3.0
Prelude>
```

---

**Exercise 2:**

1. Define a function to calculate the square of a number:

**Code**:

```haskell
square :: Int -> Int
square x = x * x

main :: IO ()
main = print (square 5)
```

**Output**:

```
subramanian@arch:~$ nvim ex2.hs
subramanian@arch:~$ ghc -o ex2 ex2.hs
[1 of 1] Compiling Main             ( ex2.hs, ex2.o )
Linking ex2 ...
subramanian@arch:~$ ./ex2
25
subramanian@arch:~$
```

---

## 2. Defining and Using Lists

Understand basic data structures like lists in Haskell.

**Code**:

```haskell
sumList :: [Int] -> Int
sumList [] = 0
sumList (x:xs) = x + sumList xs

main :: IO ()
main = do
    let result = sumList [1, 2, 3, 4, 5]
    print result
```

**Output**:

```
subramanian@arch:~$ nvim sum.hs
subramanian@arch:~$ ghc -o sum sum.hs
[1 of 1] Compiling Main             ( sum.hs, sum.o )
Linking sum ...
subramanian@arch:~$ ./sum
15
```

---

## 3. Pattern Matching with Lists

Write a function to check if a list is empty.

**Code**:

```haskell
isEmpty :: [a] -> Bool
isEmpty [] = True
isEmpty _  = False

main :: IO ()
main = do
    print (isEmpty [])
    print (isEmpty [1, 2, 3])
    print (isEmpty "hello")
```

**Output**:

```
subramanian@arch:~$ nvim empty.hs
subramanian@arch:~$ ghc -o empty empty.hs
[1 of 1] Compiling Main             ( empty.hs, empty.o )
Linking empty ...
subramanian@arch:~$ ./empty
True
False
False
```

---

## 4. Simple IO Operations

Write a program that asks the user for their name and prints a greeting.

**Code**:

```haskell
main :: IO ()
main = do
    putStrLn "What is your name?"
    name <- getLine
    putStrLn ("Hello, " ++ name)
```

**Output**:

```
subramanian@arch:~$ nvim name.hs
subramanian@arch:~$ ghc -o name name.hs
Linking name ...
subramanian@arch:~$ ./name
What is your name?
Subramanian R
Hello, Subramanian R
subramanian@arch:~$
```