1. Execute the following Arithmetic Operations at Prelude prompt. Carefully note down the outputs

```
Ajay kumar@Ajaykumar-PC MINGW64 ~
$ ghci
GHCi, version 9.2.8: https://www.haskell.org/ghc/  :? for help
ghci> 2+3
5
ghci> 2-3
-1
ghci> 2+(-3)
-1
ghci> 2*3
6
ghci> 2/3
0.6666666666666666
ghci> 3/2
1.5
ghci> (40*100)-3000+50/5
1010.0
ghci> (40*100-3000+50)/5
210.0
ghci> sqrt(36)/sqrt(36)+sqrt(25)
6.0
ghci> 2^3
8
ghci> 2**3
8.0
ghci> 2**3+2^3
16.0
ghci> sin(90)
0.8939966636005579
ghci> asin(90)
NaN
ghci> asin(pi/4)
0.9033391107665127
ghci> asin(pi/6)
0.5510695830994463
ghci> acos(pi/6)
1.0197267436954502
```

```
ghci> asinh(pi/4)
0.7212254887267799
ghci> acosh(pi/2)
1.0232274785475506
ghci> acosh(pi/4)
NaN
ghci> acosh(pi/2)+acosh(pi/4)
NaN
ghci> acosh(pi/2)+atanh(pi/4)+2**3
10.082533649370793
ghci>
```

2. Execute the following Logical and Comparison Operations at Prelude prompt. Carefully note down the outputs.

```
Ajay kumar@Ajaykumar-PC MINGW64 ~
$ ghci
GHCi, version 9.2.8: https://www.haskell.org/ghc/  :? for help
ghci> True && True
True
ghci> True && False
False
ghci> False && True
False
ghci> False && False
False
ghci> True || True
True
ghci> True || False
True
ghci> False || True
True
ghci> False || False
False
ghci> not(True)
False
ghci> not(False)
True
ghci> 100>3
True
ghci> 1000<3
False
ghci> 100 == 100
True
ghci> 100 /= 100
False
ghci> not(100 /= 100)
True
ghci> not(100 /= 100) && True
True
ghci> not(100 /= 100) && (100 > 3)
True
```

```
ghci> not(100 /= 100) && (100 > 3) || (2 == 3)
True
ghci> not(not(100 /= 100) && (100 > 3) || (2 == 3))
False
ghci>
Leaving GHCi.
```

3. Write a program in Haskell to display “Hello SFIT”.

Code:

```
{- Name: Ajaykumar Nadar
File:helloWorld.hs
Description: Hello World Program
-}

main::IO()
main=putStrLn "Hello World"
```

Output:

```
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/SEIT/PCPF/Lab/Exp_3
$ ghci helloWorld.hs
GHCi, version 9.2.8: https://www.haskell.org/ghc/  :? for help
[1 of 1] Compiling Main             ( helloWorld.hs, interpreted )
Ok, one module loaded.
ghci> main
Hello World
ghci>
```

4. Write a program in Haskell to add and subtract two integer numbers.

Code:

```
module Add_nums(main) where
  num1 :: Int
  num2 :: Int
  num1 = 2
  num2 = 3
  num3 = num1 + num2

  main :: IO()
  main =
    print num3
```

Output:

```
PS C:\Users\Ajay kumar\Desktop\SEIT\PCPF\Lab\Exp_3> runghc "c:\Users\Ajay
5
PS C:\Users\Ajay kumar\Desktop\SEIT\PCPF\Lab\Exp_3>
```

Code:

```
module Add_nums(main) where
  num1 :: Int
  num2 :: Int
  num1 = 2
  num2 = 3
  num3 = num1 - num2

  main :: IO()
  main =
    print num3
```

Output:

```
PS C:\Users\Ajay kumar\Desktop\SEIT\PCPF\Lab\Exp_3> runghc "c:\Users\Aj
-1
PS C:\Users\Ajay kumar\Desktop\SEIT\PCPF\Lab\Exp_3>
```

2. Write a program in Haskell to add, subtract, multiply two fractional numbers. Take the numbers from the users.

Code:

```haskell
import Text.Read (readMaybe)

add :: Float -> Float -> Float
add x y = x + y

subtractt :: Float -> Float -> Float
subtractt x y = x - y

multiply :: Float -> Float -> Float
multiply x y = x * y

main :: IO ()
main = do
  putStrLn "Enter the first number: "
  num1Str <- getLine
  let num1Float = readMaybe num1Str :: Maybe Float

  putStrLn "Enter the second number: "
  num2Str <- getLine
  let num2Float = readMaybe num2Str :: Maybe Float

  case (num1Float, num2Float) of
    (Just n1, Just n2) -> do
      let addition = add n1 n2
          subtraction = subtractt n1 n2
          multiplication = multiply n1 n2

      putStrLn ("Sum: " ++ show addition)
      putStrLn ("Difference: " ++ show subtraction)
      putStrLn ("Product: " ++ show multiplication)

    _ -> putStrLn "Invalid input"
```

Output:

```
PS C:\Users\Ajay kumar\Desktop\SEIT\PCPF\Lab\Exp_3> runghc "c:\Users\Ajay kumar\Desktop
Enter the first number:
4
Enter the second number:
3
Sum: 7.0
Difference: 1.0
Product: 12.0
PS C:\Users\Ajay kumar\Desktop\SEIT\PCPF\Lab\Exp_3>
```

3. Write a program in Haskell to evaluate the mathematical operations sqrt(36)+sqrt(25)

Code:

```
main :: IO()

main = do
  let result = sqrt(36) + sqrt(25)
  print result
```

Output:

```
PS C:\Users\Ajay kumar\Desktop\SEIT\PCPF\Lab\Exp_3> runghc "c:\Users
11.0
PS C:\Users\Ajay kumar\Desktop\SEIT\PCPF\Lab\Exp_3>
```