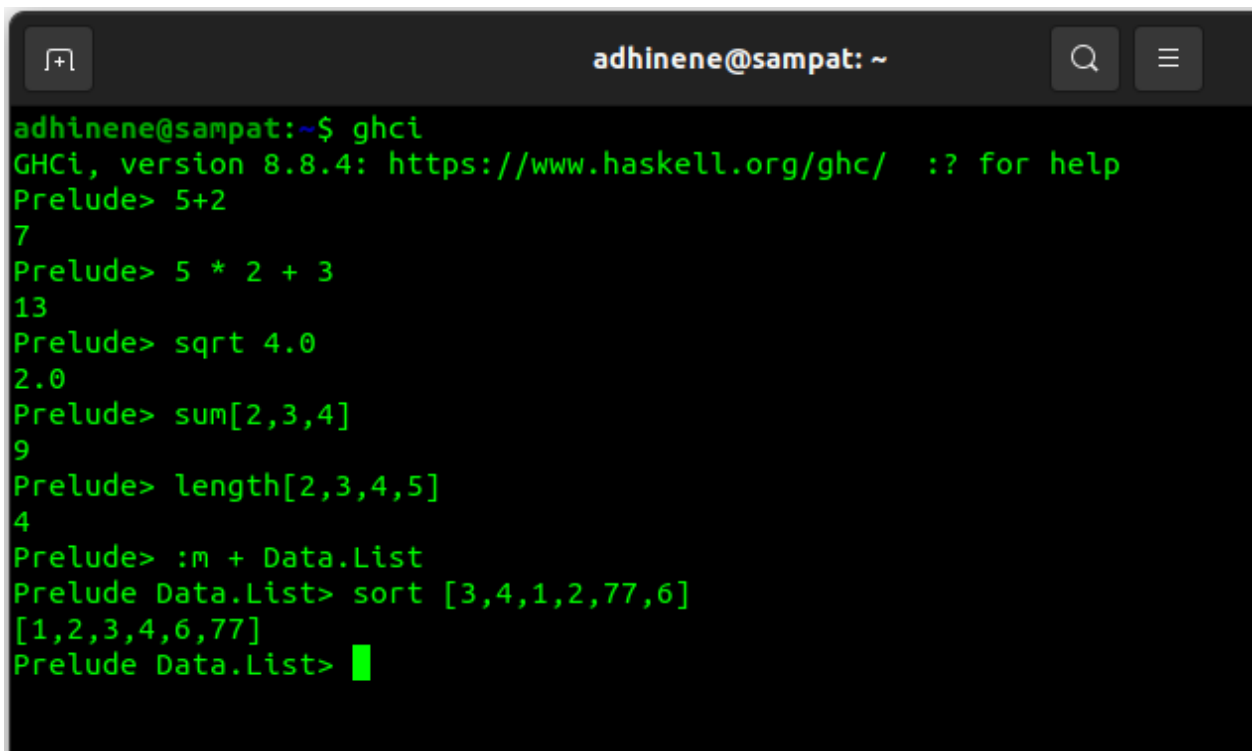


LABSHEET - 1

NAME : J.SAMPAT SRIVATSAV

ROLL NUMBER : AM.EN.U4CSE19125

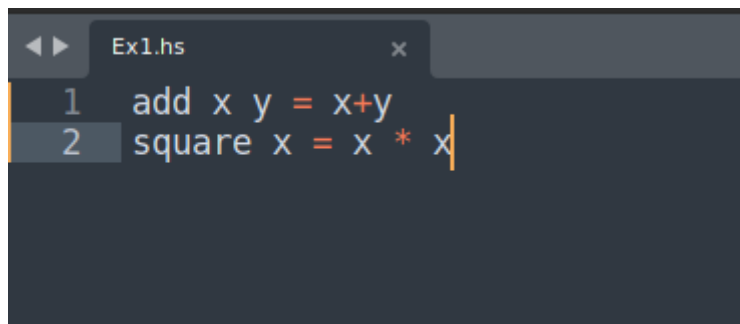
BATCH : C.S.E - B



A screenshot of a terminal window titled 'adhinene@sampat: ~'. The window shows a Haskell GHCi session. The user has entered several commands and received outputs: '5+2' returns '7', '5 * 2 + 3' returns '13', 'sqrt 4.0' returns '2.0', 'sum[2,3,4]' returns '9', 'length[2,3,4,5]' returns '4'. Then, the user imports 'Data.List' and sorts the list '[3,4,1,2,77,6]', which returns '[1,2,3,4,6,77]'. The cursor is at the end of the last line.

```
adhinene@sampat:~$ ghci
GHCi, version 8.8.4: https://www.haskell.org/ghc/  :? for help
Prelude> 5+2
7
Prelude> 5 * 2 + 3
13
Prelude> sqrt 4.0
2.0
Prelude> sum[2,3,4]
9
Prelude> length[2,3,4,5]
4
Prelude> :m + Data.List
Prelude Data.List> sort [3,4,1,2,77,6]
[1,2,3,4,6,77]
Prelude Data.List> █
```

First Script :



A screenshot of a code editor window titled 'Ex1.hs'. It contains two lines of Haskell code: '1 add x y = x+y' and '2 square x = x * x'. The cursor is at the end of the second line.

```
1 add x y = x+y
2 square x = x * x
```

```
adhinene@sampat: ~  
adhinene@sampat:~$ subl Ex1.hs  
adhinene@sampat:~$ ghci  
GHCi, version 8.8.4: https://www.haskell.org/ghc/  :? for help  
Prelude> add 3 4  
  
<interactive>:1:1: error:  
    • Variable not in scope: add :: Integer -> Integer -> t  
    • Perhaps you meant one of these:  
      ‘and’ (imported from Prelude), ‘odd’ (imported from Prelude)  
Prelude> :l Ex1.hs  
[1 of 1] Compiling Main                ( Ex1.hs, interpreted )  
Ok, one module loaded.  
*Main> add 3 4  
7  
*Main> 
```

Second Script (before changes):

```
~/Ex1a.hs - Sublime Text (UNREGISTERED)  
File Edit Selection Find View Goto Tools Project Preferences Help  
Ex1.hs Ex1a.hs  
1 quad x = x*x*x*x  
2 quad1 x = x^4  
3 quad2 x = square (square x)
```

```
adhinene@sampat:~$ subl Ex1a.hs
adhinene@sampat:~$ ghci
GHCi, version 8.8.4: https://www.haskell.org/ghc/  :? for help
Prelude> :a Ex1a.hs
[1 of 1] Compiling Main                ( Ex1a.hs, interpreted )

Ex1a.hs:3:11: error: Variable not in scope: square :: t0 -> t1
   |
3  | quad2 x = square (square x)
   |               ^^^^^^^

Ex1a.hs:3:19: error: Variable not in scope: square :: t -> t0
   |
3  | quad2 x = square (square x)
   |                   ^^^^^^^
Failed, no modules loaded.
Prelude> █
```

Second Script (after changes) :

```
~/Ex1a.hs - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
Ex1.hs  Ex1a.hs
1 module Ex1a where
2 import Ex1
3 quad x = x*x*x*x
4 quad1 x = x^4
5 quad2 x = square (square x)
```

```
~/Ex1.hs - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
Ex1.hs  Ex1a.hs
1 module Ex1(square) where
2 add x y = x+y
3 square x = x * x
```

```
adhinene@sampat: ~  
adhinene@sampat:~$ ghci  
GHCi, version 8.8.4: https://www.haskell.org/ghc/  :? for help  
Prelude> :a Ex1a.hs  
[1 of 2] Compiling Ex1          ( Ex1.hs, interpreted )  
[2 of 2] Compiling Ex1a        ( Ex1a.hs, interpreted )  
Ok, two modules loaded.  
*Ex1a> quad 5  
625  
*Ex1a> quad1 5  
625  
*Ex1a> quad2 5  
625  
*Ex1a> █
```

List Of Commands :

```

Prelude> 2*-3
<interactive>:6:2: error:
  • Variable not in scope: (*-) :: Integer -> Integer -> t
  • Perhaps you meant one of these:
    ‘*’ (imported from Prelude), ‘-’ (imported from Prelude),
    ‘*>’ (imported from Prelude)
Prelude> 2*(-3)
-6
Prelude> True && False
False
Prelude> False || True
True
Prelude> True && 1
<interactive>:10:9: error:
  • No instance for (Num Bool) arising from the literal ‘1’
  • In the second argument of ‘(&&)’ , namely ‘1’
    In the expression: True && 1
    In an equation for ‘it’: it = True && 1
Prelude> 1==1
True
Prelude> 2 /= 3
True
Prelude> not True
False
Prelude> 1 + (4 * 4)
17
Prelude> 1 + 4 * 4
17
Prelude> [1,2,3]
[1,2,3]
Prelude> [True,False,"testing"]
<interactive>:17:13: error:
  • Couldn't match expected type ‘Bool’ with actual type ‘[Char]’
  • In the expression: "testing"
    In the expression: [True, False, "testing"]
    In an equation for ‘it’: it = [True, False, "testing"]
Prelude> [1..10]
[1,2,3,4,5,6,7,8,9,10]
Prelude> [1.0,1.25..2.0]
[1.0,1.25,1.5,1.75,2.0]
Prelude> [1,4..15]
[1,4,7,10,13]
Prelude> [10,9..1]
[10,9,8,7,6,5,4,3,2,1]
Prelude> [3,1,3] ++ [3,7]
[3,1,3,3,7]
Prelude> [] ++ [False,True] ++ [True]
[False,True,True]
Prelude> 1 : [2,3]
[1,2,3]
Prelude> "This is a string"
"This is a string"
Prelude> putStrLn "Here's a newline -->\n<-- See?"
Here's a newline -->
<-- See?
Prelude> "" == []
True
Prelude> :type 3 + 2
3 + 2 :: Num a => a
Prelude> █

```