19CSE313

Principles of Programming Languages

Lab 1

S Abhishek

AM.EN.U4CSE19147

```
Hugs> 2 * -3
-6
Hugs> True && False
False
Hugs> False || True
True
Hugs> True && 1
ERROR - Cannot infer instance
*** Instance   : Num Bool
*** Expression : True && 1

Hugs> 1 == 1
True
Hugs> 2 /= 3
True
Hugs> not True
False
Hugs> 1 + (4 * 4)
17
Hugs> 1 + 4 * 4
17
Hugs> [1, 2, 3]
[1,2,3]
Hugs> [True, False, "testing"]
ERROR - Type error in list
*** Expression     : [True,False,"testing"]
*** Term           : "testing"
*** Type           : String
*** Does not match : Bool
```

```
Hugs> [1..10]
[1,2,3,4,5,6,7,8,9,10]
Hugs> [1.0,1.25..2.0]
[1.0,1.25,1.5,1.75,2.0]
Hugs> [1,4..15]
[1,4,7,10,13]
Hugs> [10,9..1]
[10,9,8,7,6,5,4,3,2,1]
Hugs> [3,1,3] ++ [3,7]
[3,1,3,3,7]
Hugs> [] ++ [False,True] ++ [True]
[False,True,True]
Hugs> 1 : [2,3]
[1,2,3]
Hugs> "This is a string."
"This is a string."
Hugs> putStrLn "Here's a newline -->\n<-- See?"
Here's a newline -->
<-- See?

Hugs> "" == []
True
Hugs> :type 3 + 2
3 + 2 :: Num a => a
```

```
Prelude Data.List> :l Ex1.hs
[1 of 1] Compiling Ex1              ( Ex1.hs, interpreted )
Ok, one module loaded.
*Ex1 Data.List> add 3 4
7
```

```
Prelude Data.List> :m + Data.List
Prelude Data.List> sort [ 4,1,6,7]
[1,4,6,7]
```

Thankyou!!