**Department of Applied Mathematics and Computational Sciences**

**PSG College of Technology**

**Programme: MSc SS VIII semester**

**Course: Functional Programming Lab**

**20PW16**

**PROBLEM SHEET 1**

**(exercises based on Programming in Haskell, Graham Hutton)**

**<u>INTRODUCTION</u>**

1. Give another possible calculation for the result of double (double 2).

```
double :: Int -> Int
double x = x + x

main = print (double (double (2)))
```

**double(2+2)**
**(2+2)+(2+2)**
**4+4**
**8**

2. Show that sum[x] = x for any number x.

```
arr = [1, 2, 3]
main = print $ 4 + sum [3, 7]

-- sum [x]
-- x + sum []
-- x + 0
-- x
```

3. Define a function product that produces the product of a list of numbers, and show using your definition that product [2, 3, 4] = 24.sum

```haskell
product' :: [Int] -> Int

product' [] = 1
product' (x : arr) = x * product' arr

main = do
    print $ product' [2, 3, 4]
```

4. How should the definition of the function *qsort* be modified so that it produces a *reverse* sorted version of a list?

```haskell
qsort [] = []

qsort (x:arr) = qsort left ++ [x] ++ qsort right
    where
        left = [i | i <- arr, i >= x]
        right = [i | i <- arr, i < x]

main = print (qsort([8, 2, 2, 3, 1, 1]))
```

5. What would be the effect of replacing ≤ by < in the definition of *qsort*? Hint: consider the example *qsort* [2, 2, 3, 1, 1].

```haskell
qsort [] = []

qsort (x:arr) = qsort left ++ [x] ++ qsort right
    where
        left = [i | i <- arr, i < x]
        right = [i | i <- arr, i > x]

main = print (qsort([2, 2, 3, 1, 1]))

--output [1, 2, 3]
```

**FIRST STEPS**

6. Parenthesise the following arithmetic expressions:

2 ↑ 3 x 4
2 x 3 + 4 x 5
2 + 3 x 4 ↑ 5

```haskell
main = do
  print((2 ^ 3) * 4)
  print((2 * 3) + (4 * 5))
  print(2 + (3 * (4 ^ 5)))
```

7. Work through the examples from this chapter using Hugs.

8. The script below contains three syntactic errors. Correct these errors and then check that your script works properly using Hugs.

   N = a 'div' length xs
   where
   a = 10
   xs = [1, 2, 3, 4, 5]

```haskell
n = a `div` length xs
  where
    a = 10
    xs = [1, 2, 3, 4, 5]

main = do
  print(n)
```

9. Show how the library function last that selects the last element of a nonempty list could be defined in terms of the library functions introduced in this chapter. Can you think of another possible definition?

```haskell
getLast1 (x) = head (reverse x)
getLast2 (x) = last x

main :: IO()
main = do
  arr <- return [1, 2, 3, 4, 5, 6]
  print (getLast1 arr)
  print (getLast2 arr)
```

10. Show how the library function init that removes the last element from a non-empty list could similarly be defined in two different ways.

```haskell
removeLast1 x = take (length x - 1) x

removeLast2 x = reverse (tail (reverse x))

main = do
  arr <- return [1, 2, 3, 4, 5]
  print(removeLast1(arr))
  print(removeLast2(arr))
```