



AMRITA
VISHWA VIDYAPEETHAM

23CSE212 – Principle of Functional Languages

BTech Computer Science and Engineering

Semester – 4

Lab Report

Academic Term: Jan – July 2025

Submitted by:

AM.SC.U4CSE23305

AMAN V SHAFEEQ

Lab Report

23CSE212 – Principles of Programming Languages

Criteria	Excellent	Good	Poor
Timely Submission			
Correctness of lab assignment			
Total Marks			
Signed By Lab Instructor			

Lab Session No: 5	
Date: 11/3/25	
Question 1: Generate a list of squares of numbers from 1 to 5 CO 2	
Code	Testcases (Input & Output)
<code>[x*x x<-[1..5]]</code>	<code>[1,4,9,16,25]</code>
Question 2: Genrate a list of even numbers from m to n	
Code	Testcases (Input & Output)
<code>[x*x x<-[1..5], mod x 2 == 0]</code>	<code>[4,16]</code>
Question 3: Generate a list of first n odd numbers CO2	
Code	Testcases (Input & Output)
<code>n_odd :: Int->[Int]</code> <code>n_odd n = take n [x x<-[1..], mod x 2 /= 0]</code>	<code>*Main> n_odd 5</code> <code>[1,3,5,7,9]</code>
Question 4: Generate a list of numbers from m to n that are divisible by 3 CO 2	
Code	Testcases (Input & Output)
<code>div_three :: Int -> Int -> [Int]</code> <code>div_three m n = [x x<-[m..n], mod x 3 ==0]</code>	<code>*Main> div_three 1 10</code> <code>[3,6,9]</code>
Question 5: Generate all ordered pairs (x,y) from x if from xs and y is from ys lists CO 2	
Code	Testcases (Input & Output)

Lab Report

23CSE212 – Principles of Programming Languages

ordered_pairs :: [Int] -> [Int] -> [(Int,Int)] ordered_pairs xs ys = [(x,y) x<-xs, y<-ys]	<pre>*Main> ordered_pairs [1,2,3] [5,6,7] [(1,5),(1,6),(1,7),(2,5),(2,6),(2,7),(3,5),(3,6),(3,7)]</pre>
Question 6: Generate all pairs (x,y) from xs and ys but only include pairs where x+y is even CO 2	
Code	Testcases (Input & Output)
ordered_even :: [Int] -> [Int] -> [(Int,Int)] ordered_even xs ys = [(x,y) x<-xs, y<-ys, (x+y) `mod` 2 == 0]	<pre>*Main> ordered_even [1,2,3] [5,6,7] [(1,5),(1,7),(2,6),(3,5),(3,7)]</pre>
Question 7: Flatten a nested list CO 2	
Code	Testcases (Input & Output)
flatten :: [[Int]]->[Int] flatten xs = foldr (++) [] xs	<pre>*Main> flatten [[1,2],[4,3]] [1,2,4,3]</pre>
Question 8: Define a function duplicate which will duplicate the list 3 times and produce a new list CO 2	
Code	Testcases (Input & Output)
pyth_triplet :: Int->Int->[[Int]] pyth_triplet m n = [(x,y,z) x<-[m..n], y<-[m..n], z<-[m..n], x^2 + y^2==z^2]	<pre>*Main> pyth_triplet 1 20 [[3,4,5],[4,3,5],[5,12,13],[6,8,10],[8,6,10],[8,15,17],[9,12,15],[12,5,13],[12,9,15],[12,16,20],[15,8,17],[16,12,20]]</pre>
Question 9 CO 2	
Code	Testcases (Input & Output)
odd_even_digits = [x x<-[10..99], (x mod 10) mod 2 == 0 && (x div 10) mod 2 /= 0]	<pre>*Main> odd_even_digits [10,12,14,16,18,30,32,34,36,38,50,52,54,56,58,70,72,74,76,78,90,92,94,96,98]</pre>
Question 10 CO 2	
Code	Testcases (Input & Output)
extract_dig :: String->String extract_dig xs = [x x<-xs, isDigit x == True]	<pre>*Main> extract_dig "Hello2134h45" "213445"</pre>
Question 11 CO 2	
Code	Testcases (Input & Output)
nested_list_sum :: [[Int]]->[Int] nested_list_sum xss = [sum xs xs<-xss]	<pre>*Main> nested_list_sum [[1,2,3],[4,5],[6,7,8,9]] [6,9,30]</pre>
Question 12 CO 2	
Code	Testcases (Input & Output)
triplets :: [Int]->Int->[(Int, Int, Int)] triplets xs s = [(x,y,z) x<-xs, y<-xs, z<-xs, x+y+z == s]	<pre>*Main> triplets [1..5] 10 [(1,4,5),(1,5,4),(2,3,5),(2,4,4),(2,5,3),(3,2,5),(3,3,4),(3,4,3),(3,5,2),(4,1,5),(4,2,4),(4,3,3),(4,4,2),(4,5,1),(5,1,4),(5,2,3),(5,3,2),(5,4,1)]</pre>
Question 13: CO 2	

Lab Report

23CSE212 – Principles of Programming Languages

Code	Testcases (Input & Output)
<div>divisors :: Int->[Int]</div> <div>divisors n = [x x<-[1..n], n `mod` x == 0]</div>	<pre>*Main> divisors 12 [1,2,3,4,6,12]</pre>
Question 14: CO 2	
Code	Testcases (Input & Output)
<div>isPrime k = if k > 1 then null [x x <- [2..k - 1], k `mod` x == 0] else False</div> <div>factors :: Int->[Int]</div> <div>factors n = [x x<-[1..n], n `mod` x == 0 && isPrime x == True]</div>	<pre>*Main> factors 28 [2,7]</pre>
Question 15: CO 2	
Code	Testcases (Input & Output)
<div>IsPrime_148 k = if k > 1 then null [x x <- [2..k - 1], k `mod` x == 0] else False</div>	<pre>ghci> isPrime_148 7 True ghci> isPrime_148 10 False</pre>

Question 16: CO 2	
Code	Testcases (Input & Output)
<div>extractVowels_148 :: [String] -> [String]</div> <div>extractVowels_148 words = [[c c <- word, c `elem` "aeiouAEIOU"] word <- words]</div>	<pre>ghci> extractVowels_148 ["Haskell","Functional","Magic"] ["ae","uioa","ai"]</pre>
Question 17: CO 2	
Code	Testcases (Input & Output)
<div>cartesianProduct_148 :: Int -> Int -> [(Int, Int)]</div> <div>cartesianProduct_148 m n = [(x, y) x <- [1..m], y <- [1..n]]</div>	<pre>ghci> cartesianProduct_148 2 3 [(1,1),(1,2),(1,3),(2,1),(2,2),(2,3)]</pre>
Question 18: CO 2	
Code	Testcases (Input & Output)
<div>multiplicationTable_148 :: Int -> [Int]</div> <div>multiplicationTable_148 n = [n * x x <- [1..10]]</div>	<pre>ghci> multiplicationTable_148 5 [5,10,15,20,25,30,35,40,45,50] ghci> multiplicationTable_148 6 [6,12,18,24,30,36,42,48,54,60]</pre>
Question 19: CO 2	
Code	Testcases (Input & Output)

Lab Report

23CSE212 – Principles of Programming Languages

<pre>triangularNumbers_148 :: Int -> [Int] triangularNumbers_148 n = [sum [1..x] x <- [1..n]]</pre>	<pre>ghci> triangularNumbers_148 5 [1,3,6,10,15] ghci> triangularNumbers_148 10 [1,3,6,10,15,21,28,36,45,55]</pre>
Question 20: CO 2	
Code	Testcases (Input & Output)
<pre>commonElements_148 :: Eq a => [a] -> [a] -> [a] commonElements_148 xs ys = [x x <- xs, x `elem` ys]</pre>	<pre>ghci> commonElements_148 [1,2,3,4,5] [3,4,5,6,7] [3,4,5] ghci> commonElements_148 [1,2,11,5] [3,11,7] [11]</pre>
Question 21: CO 2	
Code	Testcases (Input & Output)
<pre>sumPairs_148 :: [Int] -> [Int] -> [Int] sumPairs_148 xs ys = [x + y (x, y) <- zip xs ys]</pre>	<pre>ghci> sumPairs_148 [1,2,3] [4,5,6] [5,7,9] ghci> sumPairs_148 [10,2,30] [4,50,6] [14,52,36]</pre>
Question 22: CO 2	
Code	Testcases (Input & Output)
<pre>multiplyPairs_148 :: [Int] -> [Int] -> [Int] multiplyPairs_148 xs ys = [x * y (x, y) <- zip xs ys]</pre>	<pre>ghci> multiplyPairs_148 [1,2,3] [4,5,6] [4,10,18] ghci> multiplyPairs_148 [5,2,6] [10,7,9] [50,14,54]</pre>
Question 23: CO 2	
Code	Testcases (Input & Output)
<pre>pairConsecutive_148 :: [a] -> [(a, a)] pairConsecutive_148 xs = zip xs (tail xs)</pre>	<pre>ghci> pairConsecutive_148 [1,2,3,4,5] [(1,2),(2,3),(3,4),(4,5)] ghci> pairConsecutive_148 [3,4,5,6] [(3,4),(4,5),(5,6)]</pre>
Question 24: CO 2	
Code	Testcases (Input & Output)
<pre>differences_148 :: [Int] -> [Int] differences_148 xs = [abs (x - y) (x, y) <- zip xs (tail xs)]</pre>	<pre>ghci> differences_148 [1,2,15,30,25] [1,13,15,5] ghci> differences_148 [1,2,135,30,25] [1,133,105,5]</pre>
Question 20: CO 2	
Code	Testcases (Input & Output)

Lab Report

23CSE212 – Principles of Programming Languages

```
reversePairs_148 :: [a] -> [(a, a)]  
reversePairs_148 xs = zip xs (reverse xs)
```

```
ghci> reversePairs_148 [1,2,3,4,5]  
[(1,5),(2,4),(3,3),(4,2),(5,1)]  
ghci> reversePairs_148 [1,2,4,5]  
[(1,5),(2,4),(4,2),(5,1)]
```