

# COMP105 Class Test

Worth 25% of total marks for the module

TIME ALLOWED : 50 minutes

Electronic devices are not permitted

Answer all questions. Answers should be filled in on the computer-readable answer sheet.

## Section A – Recursion

1. The function `t` is defined as follows.

```
t 0 = 0
t 1 = 1
t 2 = 2
t n = t (n-3)
```

What is the result of the following query?

```
ghci> t 7
```

A. 2

B. 3

C. 0

t 7 = t 4 = t 1 = 1 -> D

D. 1

E. The query results in an infinite loop.

2. The function `f` is defined as follows.

```
f [] = []
f (x:xs) = x : (f xs) ++ [x]
```

What is the result of the following query?

```
ghci> f [1,2,3]
```

A. [1,2,3,1,2,3]

B. [1,2,3,3,2,1]

C. [3,2,1,3,2,1]

D. [3,2,1,1,2,3]

E. The query results in an error.

f [1,2,3] = 1: f[2,3] ++ "1" f [2,3] = 2:f[3] ++ "2" f [3] = 3:f[] ++ "3" = [3] ++ [2,3] ++ [1,2,3] = [3,2,3,1,2,3]

3. The function `g` is defined as follows.

```
g []      = []  
g [x]     = [x]  
g (x:y:xs) = y : g xs
```

What is the result of the following query?

```
ghci> g "abcd"
```

- A. "bd"
- B. "bc"
- C. "ac"
- D. "ad"

E. The query results in an error.

```
g "abcd" = b : g "cd" g "cd" = d:[] "bd"
```

4. Using the same definition of `g` as given in Question 3, what is the result of the following query?

```
ghci> g [1,2,3,4,5]
```

- A. [2,4]
- B. [1,3]
- C. [2,4,5]
- D. [1,3,5]

E. The query results in an error.

```
g [1,2,3,4,5] = 2:g [3,4,5] g [3,4,5]
```

5. The function `h` is defined as follows.

```
h []      acc = acc  
h (x:xs) acc = h xs (acc + 2 * x)
```

What is the result of the following query?

```
ghci> h [1,1,1,1] 0
```

- A. 15
- B. 4
- C. 8
- D. 30

E. The query results in an error.

6. The function `h` given in Question 5 is an example of:

- A. Mutual recursion.
- B. List recursion.
- C. Tail recursion.
- D. Lazy evaluation.
- E. Multiple recursion.

7. The function `p` is defined as follows.

```
p [] = 0
p (x:xs) = x + p xs + p xs
```

What is the result of the following query?

```
ghci> p [3,2,1]
```

- A. 9
- B. 17
- C. 11
- D. 6
- E. The query results in an error.

## Section B – Higher order functions

8. What is the result of the following query?

```
ghci> map (\ (x,y) -> y ) [(1,2), (3,4), (5,6)]
```

- A. [(2,1), (4,3), (6,5)]
- B. [3,7,11]
- C. [1,3,5]
- D. [2,4,6]
- E. The query results in an error.

9. What is the result of the following query?

```
ghci> filter (\ x -> length x <= 2 ) ["a", "ab", "abc", "abcd"]
```

- A. ["a", "ab"]
- B. ["a"]
- C. ["a", "ab", "abc"]
- D. ["a", "ab", "abc", "abcd"]
- E. The query results in an error.

10. What is the result of the following query?

```
ghci> foldr (\ x acc -> x ) 0 [1,2,3,4]
```

A. 1

B. 0

C. 4

D. 10

E. The query results in an error.

11. What is the result of the following query?

```
ghci> scanl1 (\ acc x -> acc ) [1,2,3,4]
```

A. [4,3,2,1]

B. [1,1,1,1]

C. [1,2,3,4]

D. [4,4,4,4]

E. The query results in an error.

12. The functions `d` and `d_list` are defined as follows.

```
d (x, y) = (x `div` 2, y / 2)
```

```
d_list xs ys = map d (zip xs ys)
```

What is the result of the following query?

```
ghci> d_list [10, 20] [4, 8]
```

A. [(5,2),(10.0,4.0)]

B. [(5,10),(2.0,4.0)]

C. [(5,2.0),(10,4.0)]

D. [(5.0,10.0),(2,4)]

E. The query results in an error.

13. What is the *most general* type annotation for the function `d` from Question 12?

A. `(Fractional a, Integral b) => (a, b) -> (a, b)`

B. `(Integral a, Fractional b) => (a, b) -> (a, b)`

C. `(Int, Float) -> (Int, Float)`

D. `(Integer, Double) -> (Integer, Double)`

E. `(a, b) -> (a, b)`

14. What is the result of the following query?

```
ghci> map (+1) . filter (<2) $ [1,2,3,4]
```

- A. [1,2]
- B. [1]
- C. [2,3]
- D. [2]
- E. []

15. The function `curry` is defined in the following way.

```
curry f = (\ x y -> f (x, y) )
```

What is the type of `curry`?

- A.  $(a \rightarrow a \rightarrow a) \rightarrow ((a, a) \rightarrow a)$
- B.  $((a, b) \rightarrow c) \rightarrow (a \rightarrow b \rightarrow c)$
- C.  $(a \rightarrow b \rightarrow c) \rightarrow ((a, b) \rightarrow c)$
- D.  $((a, a) \rightarrow a) \rightarrow (a \rightarrow a \rightarrow a)$
- E. The function will cause a compilation error.

### Section C – Custom types

16. The following custom data type will be used in Questions 16 and 17.

```
data Shape = Circle | Square | Triangle deriving (Show, Eq, Ord, Read)
```

What is the result of the following query?

```
ghci> Circle < Square && Square < Triangle
```

- A. True
- B. Circle
- C. False
- D. Triangle
- E. The query results in an error.

17. What is the result of the following query?

```
ghci> read "Triangle" :: Int
```

- A. 2
- B. 3
- C. Triangle
- D. "Triangle"
- E. The query results in an error.

18. Consider the following custom type.

```
data Point a = Point a a deriving Show
```

Which of the following points would produce an error, if typed into ghci?

- A. Point (1, 1) (2, 2)
- B. Point True False
- C. Point 1 3
- D. Point "hi" "there"
- E. Point 'a' "b"

19. The function add\_maybes is defined in the following way.

```
add_maybes (Just x) (Just y) = Just (x + y)
add_maybes Nothing  Nothing  = Nothing
```

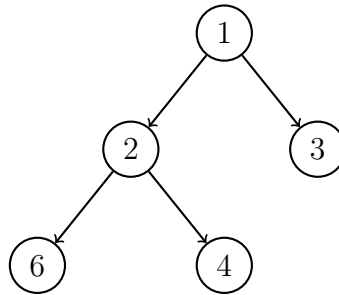
What is the result of the following query?

```
add_maybes (Just 3) Nothing
```

- A. 3
- B. Maybe 3
- C. Just 3
- D. Nothing
- E. The query results in an error.

20. Consider the following custom data-tree type.

```
data DTree a = Leaf a | Branch a (DTree a) (DTree a) deriving Show
```



The tree above can be represented as a `DTree Int` in `ghci` like so

```
ghci> let tree = Branch 1 (Branch 2 (Leaf 6) (Leaf 4)) (Leaf 3)
```

Suppose that we have loaded the following function into `ghci`.

```
tree_f (Leaf x)      = x
tree_f (Branch x l r) = tree_f l + tree_f r
```

What is the result of the following query?

```
ghci> tree_f tree
```

- A. 10
- B. 16
- C. 7
- D. 13
- E. 3

## Section D – General questions

21. Suppose that you are programming in an imperative language, and you are using a one-argument subroutine called `sub`. You call `sub` with the argument `"hello"` and it returns the integer 3. You call `sub` with the argument `"hello"` a second time, and it returns the integer 9. What can you conclude about `sub`?
- A. `sub` has no side effects.
  - B. `sub` is a pure function.
  - C. `sub` is deterministic.
  - D. `sub` is not deterministic.
  - E. None of the above are true for `sub`

22. You now call the zero-argument subroutine `open`. You observe that `open` opens a connection to an external webserver, and always returns the integer 0. Which of the following statements is true?
- A. `open` is a pure function, because it is deterministic and has no side effects.
  - B. `open` is a pure function, because it is deterministic.
  - C. `open` is a pure function, because it has no side effects.
  - D. `open` is not a pure function, because it is not deterministic.
  - E. `open` is not a pure function, because it has side effects.
23. The IO action `act` is defined as follows.

```
act :: IO Int
act = do
  x <- return 1
  y <- return 2
  z <- return 3
  return y
```

What is *returned by* the following query?

```
ghci> act
```

- A. 1
  - B. 2
  - C. IO 1
  - D. IO 2
  - E. The query produces an error.
24. Using the action `act` from Question 23, what is the result of the following query.

```
ghci> act + 2
```

- A. 4
- B. IO 4
- C. IO 3
- D. 3
- E. The query produces an error.



**25.** Consider the following function:

```
mystery = 0 : 1 : zipWith (+) mystery (tail mystery)
```

What is the result of the following query?

```
ghci> take 6 mystery
```

- A. [0,1,1,2,3,5]
- B. [0,1,0,2,0,4]
- C. [0,1,2,3,5,8]
- D. The query produces an error.
- E. The query enters an infinite loop.

Do not turn this over until the start of the test.