CS2104 Programming Language Concepts
TIC2701 Principles of Programming Languages

October 2019

You may bring in 2-pages of notes. The paper comprise 40 MCQ questions. Use a B2 pencil to answer them in the provided MCQ form, and one last question which requires written answer.

Write your matrix card number in the box below.

Section A comprises 40 multiple choice questions with 5 possible answers each. The last option, None of the Above, should be chosen if the answer cannot be found in the first four choices. (50 marks)

Consider the following code fragment in Haskell.

```
let x=0 in
let x=(let y=2 in y+x)
in x*x
```

Question 1: What result would be returned by the above code fragment?

1 [A] 4

1 [B] 5

1 [C] 6

1 [D] 7

1 [E] None of the Above

Consider the following code fragment in Haskell.

```
let x=2 in
let y=2+x in
y+x
```

Question 2: What result would be returned by the above code fragment?

2 [A] 4

2 [B] 5

2 [C] 6

2 [D] 7

2 [E] None of the Above

Consider the following function in Haskell.

```
t f x = f (f (f x))
```

Question 3: What result would be returned by `t (t (\x -> 1+x)) 1`?

3 [A] 7

3 [B] 10

3 [C] 15

3 [D] 20

3 [E] None of the Above

Question 4: What result would be returned by `t (t (\x -> 2+x)) 1`?

4 [A] 7

4 [B] 10

4 [C] 15

4 [D] 20

4 [E] None of the Above

Question 5: Given some `f`, what is (t f) equivalent to?

5 [A] `\ x -> x`

5 [B] `\ x -> f x`

5 [C] `\ x -> f (f x)`

5 [D] `\ x -> f (f (f x))`

5 [E] None of the Above

Consider the following functions in Haskell.

```
q f x = f (f x)
r     = q q
```

Question 6: What result would be returned by `r (q (\x -> 1+x)) 2`?

6 [A] 7

6 [B] 10

6 [C] 15

6 [D] 20

6 [E] None of the Above

Question 7: What result would be returned by `q (r (\x -> 1+x)) 2`?

7 [A] 7

7 [B] 10

7 [C] 15

7 [D] 20

7 [E] None of the Above

Question 8: What is the most general type for function `r`?

8 [A] `a -> b -> c`

8 [B] `a -> (a -> b) -> b`

8 [C] `a -> (a -> a) -> a`

8 [D] `a -> ((a -> a) -> a) -> a`

8 [E] None of the Above

Question 9: What result would be returned by `(f 10 5)` under below?

```
f x a = if x==0 then a
        else f (x-1) (a+2)
```

9 [A] 15

9 [B] 20

9 [C] 25

9 [D] 30

9 [E] None of the Above

Consider the following function in Haskell.

```
g x = if x==0 then 0
      else 2 + g (x-1)
```

Question 10: What result would be returned by `(g 15)`?

10 [A] 15

10 [B] 20

10 [C] 25

10 [D] 30

10 [E] None of the Above

Question 11: What result would be returned by `(g (-1))`?

11 [A] 15

11 [B] 20

11 [C] 25

11 [D] 30

11 [E] None of the Above

Consider the following function in Haskell.

```
h n x y z =
   if (n::Int)==0 then (x,y,z)
   else h (n-1) y z x
```

Question 12: What result would be returned by `(h 5 1 2 3)`?

12 [A] (1,2,3)

12 [B] (2,3,1)

12 [C] (3,1,2)

12 [D] (3,2,1)

12 [E] None of the Above

Question 13: What result would be returned by `(h 50 1 2 3)`?

13 [A] (1,2,3)

13 [B] (2,3,1)

13 [C] (3,1,2)

13 [D] (3,2,1)

13 [E] None of the Above

Question 14: What result would be returned by `(h 2 1 'a' 3)`?

14 [A] `(1,'a',3)`

14 [B] `('a',3,1)`

14 [C] `(3,1,'a')`

14 [D] `(3,'a',1)`

14 [E] None of the Above

Consider the following function in Haskell.

```
h2 f r xs =
   case xs of
     [] -> r
     (y:ys) -> (f y):(h2 f r ys)
```

Question 15: What result would be returned by `(h2 (+1) [1,2,3] [9])`?

15 [A] `[1,2,3,9]`

15 [B] `[2,3,4,9]`

15 [C] `[10,1,2,3]`

15 [D] `[10,2,3,4]`

15 [E] None of the Above

Question 16: What result would be returned by `(h2 (+1) [9] [1,2,3])`?

16 [A] `[1,2,3,9]`

16 [B] `[2,3,4,9]`

16 [C] `[10,1,2,3]`

16 [D] `[10,2,3,4]`

16 [E] None of the Above

Question 17: What is the most general type for method `(h2)`?

17 [A] `(a -> b) -> [a] -> [b] -> [b]`

17 [B] `(a -> b) -> [b] -> [a] -> [b]`

17 [C] `(a -> b) -> a -> b -> b`

17 [D] `(a -> b) -> [a] -> b -> [b]`

17 [E] None of the Above

Consider the following function in Haskell.

```
h3 f x =
    let a = (f x)
    in a : (h3 f a)
```

Question 18: What result would be returned by `length(h3 (+1) 0)`?

18 [A] `0`

18 [B] `2`

18 [C] `5`

18 [D] `100`

18 [E] None of the Above

Question 19: What result would be returned by `length(take 5 (h3 (+1) 0))`?

19 [A] `0`

19 [B] `2`

19 [C] `5`

19 [D] `100`

19 [E] None of the Above

Question 20: What is the most general type for method `(h3)`?

20 [A] `(a -> b) -> a -> [b]`

20 [B] `(a -> a) -> a -> [a]`

20 [C] `a -> (a -> b) -> [b]`

20 [D] `a -> (a -> a) -> [a]`

20 [E] None of the Above

Consider the following recursive function in Haskell.

```
goo x =
  if x<=1 then 1
  else goo (x-1) + (goo (x-1))
```

Question 21: What result would be returned by the following call `(goo 4)`?

21 [A] 2

21 [B] 4

21 [C] 8

21 [D] 16

21 [E] None of the Above

Question 22: How many `goo` calls were made by `(goo 4)`? (You are to count all `goo` calls including the first call, and the base-case calls.)

22 [A] 4

22 [B] 8

22 [C] 16

22 [D] 32

22 [E] None of the Above

Question 23: What is the best and simplest way to optimize this `goo` method?

23 [A] Convert it to a tail-recursive method.

23 [B] Convert it into a loop.

23 [C] Use constraint solving to implement this method.

23 [D] Use memoization technique to eliminate redundant calls.

23 [E] Use `let` construct eliminate duplicated expressions.

Consider the following function in Haskell.

```
h4 f x = (\ f g x -> g (f x) x) (\x -> (f (1::Int)) x) f x
```

Question 24: What is the most general type for method `(h4)`?

24 [A] `(Int -> Int ->Int) -> Int -> Int`

24 [B] `(Int -> t ->Int) -> Int -> Int`

24 [C] `(Int -> t ->Int) -> t -> Int`

24 [D] `(Int -> t ->Int) -> t -> Int`

24 [E] None of the Above

Question 25: What result would be returned by the following call `(h4 (+) 2)`?

25 [A] 2

25 [B] 3

25 [C] 4

25 [D] 5

25 [E] None of the Above

Question 26: What result would be returned by the following call `(h4 (*) 2)`?

26 [A] 2

26 [B] 3

26 [C] 4

26 [D] 5

26 [E] None of the Above

Consider the following recursive function in Haskell.

```
map :: (a->b) -> [a] -> [b]
foldr :: (a -> r -> r) -> r -> [a] -> r
h5 g f r xs = foldr g r (map f xs)
```

Question 27: What result would be returned by the following call `(h5 (+) (+1) 0 [1..4])`?

27 [A] 4

27 [B] 8

27 [C] 14

27 [D] 18

27 [E] None of the Above

Question 28: How many new constructor `(:)` was need to run the code `(h5 (+) (+1) 0 [1..4])`?

28 [A] 4

28 [B] 8

28 [C] 16

28 [D] 32

28 [E] None of the Above

Question 29: What result would be returned by the following call `h5 (++) (\c -> [c]) "t" "hello"`?

29 [A] `"hello"`

29 [B] `"hheellllloo"`

29 [C] `"hellothere"`

29 [D] `"hellohello"`

29 [E] None of the Above

Consider the following recursive function in Haskell.

```
h6 f xs = foldr (&&) True ([f x| x <- xs])
g2 ys xs = [y| y<-ys, h6 (\ x->x/=y) xs]
```

Question 30: What result would be returned by the following call `(g2 "hello" "oe")`?

30 [A] `""`

30 [B] `"hell"`

30 [C] `"hllo"`

30 [D] `"hll"`

30 [E] None of the Above

Question 31: What is the inferred type for `g2` function?

31 [A] `Eq a => [a] -> [a] -> [a]`

31 [B] `Ord a => [a] -> [a] -> [a]`

31 [C] `[a] -> [b] -> [a]`

31 [D] `[a] -> [a] -> [a]`

31 [E] None of the Above

Question 32: What does expression `(g2 ys xs)` compute from the lists `ys` and `xs`?

32 [A] the union of the two lists

32 [B] the intersection of the two lists

32 [C] the difference of the two lists

32 [D] the first list `ys`

32 [E] None of the Above

Consider the following function in Haskell.

```
h7 m1 m2 = do
             x <- m1
             y <- m2
             return (x,y)
```

Question 33: What is the most general type for method `(h7)`?

33 [A] `Monad m => m a -> m a -> m (a,a)`

33 [B] `Monad m => m a -> m b -> m (a,b)`

33 [C] `Monad m => m a -> m b -> m (b,a)`

33 [D] `(Monad m1,Monad m2) => m1 a -> m2 b -> m2 (a,b)`

33 [E] None of the Above

Question 34:
What is the result of `(h7 [1,2] "ab")`?

34 [A] `[(1,'a'),(1,'b'),(2,'a'),(2,'b')]`

34 [B] `[(1,'a'),(2,'a'),(1,'b'),(2,'b')]`

34 [C] `[(1,'a'),(2,'b')]`

34 [D] `([1,2],"ab")`

34 [E] None of the Above

Question 35: What is the result of `(h7 (Just [1,2]) (Just "ab")`?

35 [A] `[(1,'a'),(1,'b'),(2,'a'),(2,'b')]`

35 [B] `[(1,'a'),(2,'a'),(1,'b'),(2,'b')]`

35 [C] `[(1,'a'),(2,'b')]`

35 [D] `([1,2],"ab")`

35 [E] None of the Above

Question 36: What is the result of `(h7 (+ (1::Int)) (* (3::Int)) 1)`?

36 [A] `(1,3)`

36 [B] `(2,3)`

36 [C] `(3,3)`

36 [D] `(3,1)`

36 [E] None of the Above

Question 37: What is expression `(h7 (+ (1::Int)) (* (3::Int)) x)` equivalent to?

37 [A] `(1,3)`

37 [B] `(1+x,3*x)`

37 [C] `(x+x+x,x*x*x)`

37 [D] `(x,3*x)`

37 [E] None of the Above

Consider the following function in Haskell.

```
hoo f1 f2 = h7 (f1 (1::Int)) (f2 (3::Int))
```

Question 38: What is the most general type of `hoo`?

38 [A] `(Int -> t -> a) -> (Int -> t -> b) -> t -> (a, b)`

38 [B] `(Int -> t -> a) -> (Int -> t -> b) -> t -> (b, a)`

38 [C] `(Int -> t -> a) -> (Int -> t -> a) -> t -> (a, a)`

38 [D] `(Int -> t -> a) -> (Int -> t -> b) -> t2 -> (a, b)`

38 [E] None of the Above

Question 39: Which of the following is a monad `Monad (m a)` used by the `hoo` function?

39 [A] `m a = [a]`

39 [B] `m a = IO a`

39 [C] `m a = b -> a`

39 [D] `m a = Int -> a`

39 [E] None of the Above

Question 40: Which of the following is NOT true about arrays in Haskell?

40 [A] Arrays are immutable in Haskell.

40 [B] Values of arrays are lazily evaluated.

40 [C] Array indexes are members of a type class.

40 [D] Arrays must be built in some particular order in Haskell.

40 [E] None of the Above

Question 41 (Monadic Implementation for `hoo`)
Implement the underlying monadic operations, `return` and `>>=`, that are required by function `hoo`. Write your answer in the space below.