Midterm4-Version B - Name Key

For all problems, choose *best* possible answer and make answer clearly known by circling or writing the letter of your answer.

1. [5]What is the introduction form for the function type in Lettuce?
   A. `FunDef`
   B. `Let`
   C. `Bind`
   D. `FunApp`

2. [5]`IfThenElse` is really just a special case of:
   A. pattern matching
   B. polymorphism
   C. function definition
   D. bind

3. [5]What is the process of determining a type of an (sub)-expression given a context?
   A. Type Checking
   B. Type Guessing
   C. Well Formedness
   D. Type Inference

4. [5]What are the three parts of a Closure?
   A. Argument, Function Body, Environment
   B. Argument, Closure, Context
   C. Parameter, Environment, Extensions
   D. Argument, Type, Dictionary

5. [5]What is the introduction form for the `num` type in Lettuce?
   A. `NumVal`
   B. $\mathbb{Z}$
   C. `Plus`
   D. `Num`

6. [5]Is Error a type in Lettuce?
   A. Yes, because maybe is a type
   B. No, It could be a constructor in the type `Type` but it does not represent an actual type in our type system, it is only used for errors when type checking
   C. Yes, Every expression has a type and we could give a type to expressions which result in errors
   D. No, it could be used as a type annotation though

7. [5]What is the elimination form for the function type in Lettuce?
   A. `FunDef`
   B. `Bind`
   C. `Let`
   D. `FunApp`

8. [5]What is the difference between `choose` and `option`?
   A. `choose` parses both inputs while `option` only uses the second parser if the first is unsuccessful
   B. `option` parses both inputs while `choose` only uses the second parser if the first is unsuccessful
   C. They are the same

9. [5]What should the ??? be replaced with in the inference rule below?

$$\frac{}{\sigma \vdash ???\ x\ e \Downarrow \text{Closure}\ x\ e\ \sigma}$$

   A. `Closure`
   B. `FunCall`
   C. `LetRec`
   D. `FunDef`

10. [5]What is the advantage of type checking over allowing for runtime errors?
   A. We gain confidence that certain classes of errors will not appear at runtime before even running the program
   B. We can get free documentation by having annotations in our program
   C. We are able to say a program will always terminate so we don't have infinite loops

11. [5]Does type checking, as we have defined it, eliminate all possibility of runtime errors if a program is well-typed?
   A. yes
   B. no

12. [5]What is should replace ??? below?
   TC-IfThenElse

$$\frac{???}{\Gamma \vdash \texttt{IfThenElse}\ e_p\ e_t\ e_f : \tau}$$

   A. $\Gamma \vdash e_p : \texttt{bool} \quad \Gamma \vdash e_t : \tau_1 \quad \Gamma \vdash e_f : \tau_2$
   B. $\Gamma \vdash e_p : \texttt{bool} \quad \Gamma \vdash e_t : \tau \quad \Gamma \vdash e_f : \tau$
   C. $\Gamma \vdash e_p : \tau \quad \Gamma \vdash e_t : \tau \quad \Gamma \vdash e_f : \tau$
   D. $\Gamma \vdash e_p : \texttt{bool} \quad \Gamma \vdash e_t : \texttt{bool} \quad \Gamma \vdash e_f : \texttt{bool}$

13. [5]What should repalce ??? in the type checking rule below?
   TC-Plus

$$\frac{???}{\Gamma \vdash \texttt{Plus}\ e_1\ e_2 : \texttt{num}}$$

   A. Nothing - There is no premise for this rule
   B. $\Gamma \vdash e_1 : \mathbb{N} \quad \Gamma \vdash e_2 : \mathbb{N}$
   C. $\Gamma \Downarrow e_1 : \texttt{num} \quad \Gamma \Downarrow e_2 : \texttt{num}$
   D. $\Gamma \vdash e_1 : \tau \quad \Gamma \vdash e_2 : \tau$
   E. $\Gamma \vdash e_1 : \texttt{num} \quad \Gamma \vdash e_2 : \texttt{num}$

14. [5]What should repalce ??? in the type checking rule below?
   TC-Let

$$\frac{\Gamma \vdash e_1 : \tau_x \quad ??? \vdash e_2 : \tau}{\Gamma \vdash \texttt{Let}\ x\ \tau_x\ e_1\ e_2 : \tau}$$

   A. $\Gamma[x \mapsto \tau]$

B. Γ
C. $\Gamma[x \mapsto \tau_x]$
D. $\Gamma(x)$

15. [5]What is the type of the following lettuce expression?

```
let x : num = 5
  in if x > 10
     then 5
     else true
```

A. `bool`
B. `num`
C. Type Error - it does not have a type
D. `num -> num`

16. [5]What should repalce `???` in the type checking rule below?
TC-NUM

$$\frac{}{\Gamma \vdash \texttt{Num}\ n : ???}$$

A. `num`
B. n
C. $\Gamma(n)$
D. Type Error

17. [5]What does the following program evaluate to?

```
let fac = function(x)
            if x >= 0
              then x * fac(x - 1)
              else 1
in fac(4)
```

A. Zero
B. 6
C. 24
D. Error

18. [5]What is the elimination form for the `bool` type in Lettuce?
A. `IfThenElse`
B. `Let`
C. `Plus`
D. `Geq`

19. [5]What is the type annotation that should replace ??? below?

```
let f : ??? = function(x : num)
                x + x
in f(y)
```

A. `num -> num -> num`
B. `num`
C. `Plus`
D. Type Error - it does not have a type
E. `num -> num`

20. [5]Why do we perform type checking?
A. In order to eliminate a class of errors before even running a program
B. In order to ensure we never lookup data from a dictionary that isn't there
C. To ensure our program always terminates
D. To eliminate integer overflow errors