Midterm2 - Draft Name ______________________________

1. [5]When a function only includes recursive calls as the *outermost* computation, we say it is:
   A. higher-order
   B. pure
   C. total
   D. tail-recursive

2. [5]Dictionaries, as we have implemented them, could also have been implemented using existing data structures. What are these?
   A. Maybe and Int
   B. List and Nat
   C. Maybe and Either
   D. List and Tuples/Pairs

3. [5]This is the *meaning* of a program in some language
   A. Context
   B. Grammar
   C. Syntax
   D. Evaluation
   E. Semantics

4. [5]This describes the symbols or notation of a programming language
   A. Inference Rules
   B. Operators
   C. Semantics
   D. Syntax

5. [5]Which of the following constructors is *not* a value which can be evaluated from a lettuce program?
   A. `Error`
   B. `BinVal`
   C. `NumVal`
   D. `String`

6. [5]The following code is an example of which kind of syntax?

```
Let 'x' (Plus (Num 5) (Bin True)) (Mult (Ident 'x') (Ident 'y'))
```

   A. Abstract Syntax
   B. Concrete Syntax

7. [5]The following code is an example of which kind of syntax?

```
Num 5
```

   A. Abstract Syntax
   B. Concrete Syntax

8. [5]The following code is an example of which kind of syntax?

```
if x > 5\n  then 4 \n  else x + 15
```

A. Abstract Syntax
B. Concrete Syntax

9. [5]The following code is an example of which kind of syntax?

```
5 * 5
```

A. Abstract Syntax
B. Concrete Syntax

10. [5]The following code is an example of which kind of syntax?

```
let x = 5 \n  in x + x
```

A. Abstract Syntax
B. Concrete Syntax

11. [5]Which is the correct way to read an inference rule?

A. $\dfrac{\textbf{Numerator}}{\textbf{Denominator}}$

B. $\dfrac{\textbf{Recursion}}{\textbf{Evaluation}}$

C. $\dfrac{\textbf{Input}}{\textbf{Output}}$

D. $\dfrac{\textbf{Premise}}{\textbf{Conclusion}}$

12. [5]Which constructor/lettuce expression should replace the question marks in the rule below?

$$\frac{\sigma \vdash e_1 \Downarrow v_1 \quad \sigma[x \mapsto v_1] \vdash e_2 \Downarrow v_2}{\sigma \vdash ???\ x\ e_1\ e_2\ \Downarrow v_2}$$

A. `Ident`
B. `Plus`
C. `Let`
D. `IfThenElse`

13. [5]Which constructor/lettuce expression should replace the question marks in the rule below?

$$\frac{x \in \sigma}{\sigma \vdash ???\ x\ \Downarrow \sigma(x)}$$

A. `Bin`
B. `Num`
C. `Let`
D. `Ident`

14. [5]Which constructor/lettuce expression should replace the question marks in the rule below?

$$\frac{\sigma \vdash e_1 \Downarrow \texttt{True} \quad \sigma \vdash e_2 \Downarrow v_2}{\sigma \vdash ???\ e_1\ e_2\ e_3\ \Downarrow v_2}$$

A. `And`
B. `Or`
C. `Let`
D. `IfThenElse`

15. [5]This type handles errors when a dictionary lookup fails because the key isn't present
A. `Dict`
B. `Environment`
C. `Bool`
D. `Int`
E. `Nat`
F. `Maybe`
G. `Value`
H. `Expr`

16. [5]This is the type of the result of a Lettuce program's computation
A. `Dict`
B. `Environment`
C. `Bool`
D. `Int`
E. `Nat`
F. `Maybe`
G. `Value`
H. `Expr`

17. [5]This type is the type of Lettuce *programs* before they have been evaluated
A. `Dict`
B. `Environment`
C. `Bool`
D. `Int`
E. `Nat`
F. `Maybe`
G. `Value`
H. `Expr`

18. [5]We used this Type to implement the environment(σ) for the Lettuce interpreter:
A. `Dict`

B. `Environment`
C. `Bool`
D. `Int`
E. `Nat`
F. `Maybe`
G. `Value`
H. `Expr`

19. [5]What is the process of ensuring our program's inputs and outputs are the right kind of thing? (For instance, if I give a number to the absolute value function, I would expect to get a number back)
A. Inference Rules
B. Denotational Semantics
C. Type Checking
D. Well-Formedness
E. Big-Step Operational Semantics

20. [5]This is the style of notation we prefer to write our evaluation rules in. It lends itself nicely to creating proof trees
A. Inference Rules
B. Denotational Semantics
C. Type Checking
D. Well-Formedness
E. Big-Step Operational Semantics

21. [5]This is the property of a lettuce program where each variable is always defined before it is used
A. Inference Rules
B. Denotational Semantics
C. Type Checking
D. Well-Formedness
E. Big-Step Operational Semantics

22. [5]What is the name of the type of semantics we have defined for evaluating Lettuce programs?
A. Inference Rules
B. Denotational Semantics
C. Type Checking
D. Well-Formedness
E. Big-Step Operational Semantics

23. [5]Evaluate the following Lettuce Expression:

```
let x = (let x = 7 in x + x) in
  let y = 13 in
      x + y
```

A. 27
B. 20
C. 33
D. 34

24. [5]Evaluate the following Lettuce Expression:

```
let x = 2 in
  let y = 9 in
    let x = 5 in
      x + y + x
```

A. 19
B. 13
C. 16
D. 20

25. [5]Evaluate the following Lettuce Expression:

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

A. 4
B. 2
C. 6
D. Error