Program: **Information Technology**

Curriculum Scheme: Rev2019

Examination: SE Semester III (DSE)

Course Code: ITC305 and Course Name: Paradigms and Computer Programming Fundamentals

Time: 2 hour Max. Marks: 80

|  |  |
| --- | --- |
| **Q1.** | **Choose the correct option for following questions. All the Questions are compulsory and carry equal marks** |
|  |  |
| 1. | Which is **NOT** a correct syntax for a type signature for a haskell binary function named “**foo”** ? |
| Option A: | foo :: a-> a-> a |
| Option B: | foo :: Num a => a -> a -> a |
| Option C: | foo :: Num a => (b->a) -> a ->a |
| Option D: | foo :: Num a => b ->a -> a -> a |
|  |  |
| 2. | Image 1 shows contents of two distinct prolog codes KB-1 and KB-2  Which of the following statements is **true** about the above two KBs |
| Option A: | Query path(a,a) will evaluate as true for both KBs |
| Option B: | Query path(a,a) will evaluate as false for both KBs |
| Option C: | Query path(a,a) will evaluate as true for KB-1 and false in KB-2 |
| Option D: | Query path(a,a) will evaluate as true for KB-1 and will not terminate in KB-2 |
|  |  |
| 3. | When parameters are passed to a subroutine while calling it , are known as \_\_\_\_\_ |
| Option A: | Formal parameters |
| Option B: | Normal parameters |
| Option C: | Actual parameters |
| Option D: | Additional parameters |
|  |  |
| 4. | Consider a list a=[1, 2, 3, 4, 5, 6, 7, 8, 9, 10] is available in Haskell's interactive environment. If we execute following statement at prelude prompt what will be the output:  let (y,z) = splitAt 1 a in y ++ (tail z) |
| Option A: | [1,2,3,4,5,6,7,8,9,10] |
| Option B: | [1,1,3,4,5,6,7,8,9,10] |
| Option C: | [1,3,4,5,6,7,8,9,10] |
| Option D: | [1,1,2,3,4,5,6,7,8,9,10] |
|  |  |
| 5. | Image 2 shows a prolog code that performs some arithmetic operations. What will be the output, if we pose queries **calculate(F, 5)** and **calculate(5,5)** separately to the prolog interpreter based on this code? |
| Option A: | false and 5 |
| Option B: | F=3 and true |
| Option C: | F=5 and true |
| Option D: | F=3 and false |
|  |  |
| 6. | Which is the incorrect query in Prolog from the following? |
| Option A: | ?- is(X, 1+2). |
| Option B: | ?- X is 1+2. |
| Option C: | ?- 1+2 is 4-1. |
| Option D: | ?- is(1+2,X). |
|  |  |
| 7. | Compiler translates high level language source code into \_\_\_\_\_ |
| Option A: | corrected code |
| Option B: | object code |
| Option C: | pre code |
| Option D: | document code |
|  |  |
| 8. | From the following statements, which is **not** true about Coroutines? |
| Option A: | Coroutines are execution contexts. |
| Option B: | Coroutines can not share a single stack. |
| Option C: | Coroutines can not be used to implement iterators. |
| Option D: | Coroutines can be used to implement threads. |
|  |  |
| 9. | Which of the following is incorrect about Haskell |
| Option A: | It follows declarative style of programming |
| Option B: | Adopts principles of lambda calculus |
| Option C: | Store the state of the function in the form of variables |
| Option D: | Includes only pure functions |
|  |  |
| 10. | Which of the following is true about polymorphism in Haskell? |
| Option A: | type variables in haskell is an instance of parametric polymorphism whereas type classes in haskell is an instance of ad-hoc polymorphism. |
| Option B: | type variables in haskell is an instance of ad-hoc polymorphism whereas type classes in haskell is an instance of parametric polymorphism. |
| Option C: | type variables and type classes in haskell are instances of parametric polymorphism. |
| Option D: | type variables and type classes in haskell are instances of ad-hoc polymorphism. |
|  |  |
| 11. | Which of the following commands tells the Prolog system to fail a particular goal immediately without trying for alternate solutions. |
| Option A: | not |
| Option B: | cut |
| Option C: | unify |
| Option D: | disjunction |
|  |  |
| 12. | Which of the following is **NOT** a Type class in Haskell. |
| Option A: | Bounded |
| Option B: | Functor |
| Option C: | Integral |
| Option D: | String |
|  |  |
| 13. | Which of the following is true for Implicit parametric polymorphism |
| Option A: | Parameter types are not specified at all and not type-safe |
| Option B: | Parameter types to be specified explicitly, but still type-safe |
| Option C: | Parameter types are incompletely specified and not type-safe |
| Option D: | Parameter types are incompletely specified, but still type-safe |
|  |  |
| 14. | From the following, which can not be considered as variable in Prolog? |
| Option A: | A |
| Option B: | \_h |
| Option C: | What |
| Option D: | x |
|  |  |
| 15. | Which of the following is used in logic programming? |
| Option A: | classes |
| Option B: | resolution and unification |
| Option C: | monad |
| Option D: | iterative constructs |
|  |  |
| 16. | When binding of the referencing environment of a subroutine that has been passed as a parameter, occurs late then it is known as \_\_\_ and which is usually default in languages with \_\_\_\_\_. |
| Option A: | Shallow binding, dynamic scoping |
| Option B: | Shallow binding, static scoping |
| Option C: | deep binding, dynamic scoping |
| Option D: | deep binding, static scoping |
|  |  |
| 17. | The period of time between the creation and the destruction of a name-to object binding is referred as |
| Option A: | binding lifetime |
| Option B: | object lifetime |
| Option C: | runtime lifetime |
| Option D: | referencing |
|  |  |
| 18. | Which of the programming language DOES NOT belongs to declarative programming paradigm |
| Option A: | XML |
| Option B: | SQL |
| Option C: | prolog |
| Option D: | java |
|  |  |
| 19. | Choose the most appropriate feature of the functional programming used in the Haskell code shown in image 4:  relate :: (c -> d) -> [c] -> [d]  relate \_ [] = []  relate f (x:xs) = f x : relate f xs  Image 4 |
| Option A: | Polymorphism |
| Option B: | Higher order function |
| Option C: | Aggregates for structured objects |
| Option D: | Garbage Collection |
|  |  |
| 20. | Maintenance of the stack is done by \_\_\_\_\_\_\_. |
| Option A: | Subroutine calling sequence / Subroutine frames |
| Option B: | Prologue2 / Subroutine local variables |
| Option C: | Epilogue / Subroutine return values |
| Option D: | Subroutine calling sequence, Prologue and Epilogue |
|  |  |

|  |  |
| --- | --- |
| **Q2.** | **Solve any Four out of Six 5 marks each** |
| A | Explain how Prolog differs from imperative languages in its handling of arithmetic. |
| B | Justify the following statement, “No single factor determines whether a programming language is good.” |
| C | Explain concept of currying in haskell with an example. |
| D | Explain what are facts, rules, and queries in logic programming with example. |
| E | The haskell function head defined in prelude, returns the first element of a list and throws an exception when we try to apply it on an empty list.  Define two variants of this function (you can use different names) that work exactly like head function except in the case of an empty list input they will show [] as output instead of throwing an exception.  You must use the following constructs in Haskell for defining the functions.   1. First implementation should make use of pattern matching. 2. Second implementation uses guard equations   **Note: Students are not expected to write the main function and do uer IO.** |
| F | Describe different parameter passing modes. |
|  |  |
| **Q3.** | **Solve any Four out of Six 5 marks each** |
| A | Compare heap based and stack based principle storage allocation mechanisms. |
| B | Write a note on Lambda Calculus. |
| C | What is the difference between normal-order and applicative-order evaluation? What is lazy evaluation? |
| D | Describe the difference between forward chaining and backward chaining. Which is used in Prolog by default? |
| E | Define a haskell function named “addUs” that adds 2 input numbers.  Using this function as a building block, define a Haskell function “multiplyUs” that multiplies two input numbers.  The multiplyUs function should cater to following:  1. Inputs may be signed numbers e.g. “multiplyUs (-2) \* (3)” should result in “-6” and “multiplyUs (-2) \* (-6)” should result in “12”  2. It should use guard expressions and recursion.  3. No need to write the main function to do user interaction writing definition for “addUs” and “multiplyUs” is sufficient. |
| F | Discuss Scope with reference to binding in program. Also compare static and dynamic scoping. |