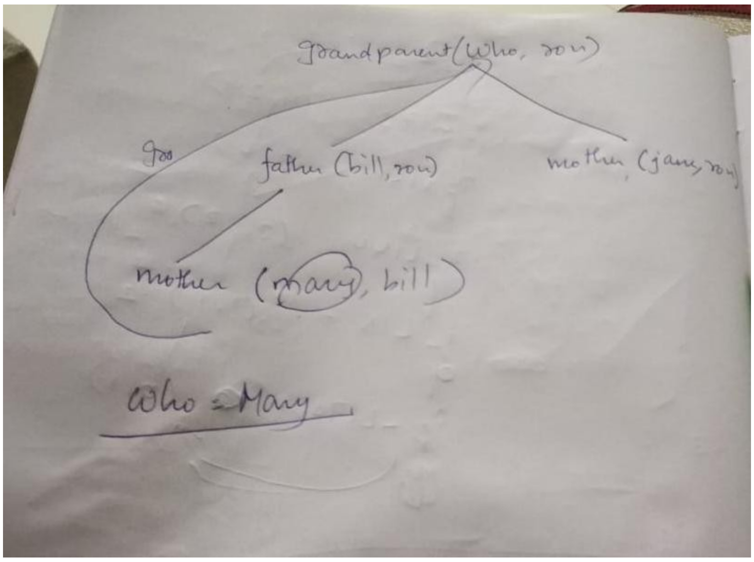
Ex 15.2

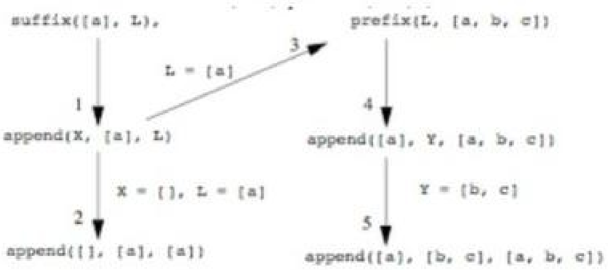
```
mammals(cow).
noarms(cow).
```

Ex 15.3

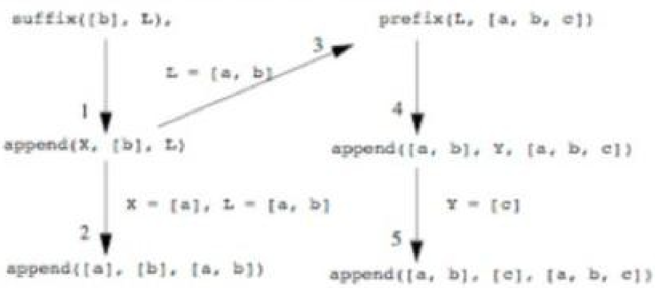


Ex 15.10

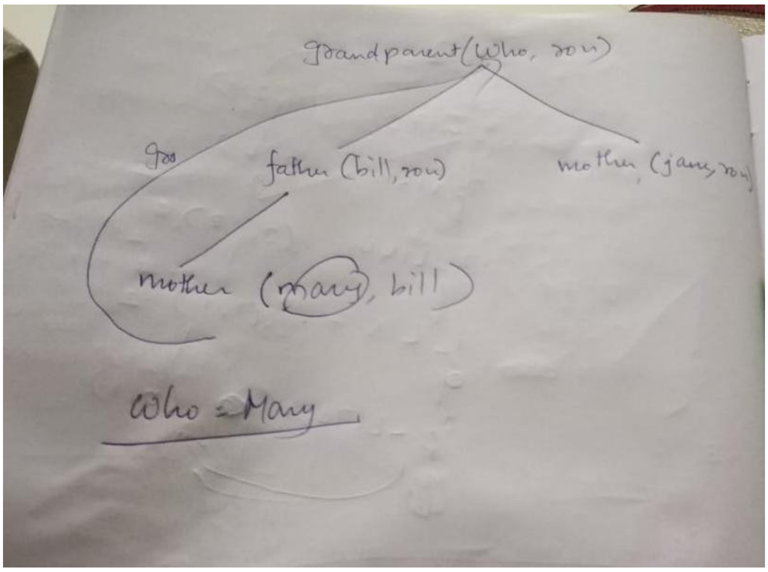
a)



b)



Ex 15.11



Ex 15.13

(a)

(1)True
(2)True
(3) False

Ex 15.21

LOGIC PROGRAMMING

Logic programming is a type of programming paradigm which is largely based on formal logic. Any program written in a logic programming language is a set of sentences in logical form, expressing facts and rules about some problem domain. Major logic programming language families include Prolog, Answer set programming (ASP) and Datalog. In all of these languages, rules are written in the form of clauses: Logic programming. In logic programming, you program by creat- ing a database of axioms. The program is executed by entering a theorem and asking the system to find a proof given the set of axioms. Prolog, the most popular exampleof this programming style, usesfirst- order predicate logic to derive theorems from the set of axioms in a database. Prolog databases are constructed of facts and rules. An ex- ample of a Prolog fact is temperature(jones,lOl ,oral) declarative reading of this statement might be “The patient Jones has an oral temperature of 101 degrees.” Prolog rules are if-then state- ments of the form fever(Patient) :- temperature(Patient,Temp,oral), Temp > 100.

This rule would read, “If a patient has an oral temperature in excess of 100 degrees, that patient has a fever.” With this pair of Prolog clauses, you could ask the system to prove a theorem that Jones has a fever. Even though there are no facts in the database about Jones or fevers, Prolog could derive the theorem using the fever rule and the single fact about Jones’s temperature.

- Availibility of inference difficult to rules
- Simple notation
- Good to understand
- Conceptual economy: every fact represented only once difficult to represent procedural knowledge
- Not well organized

OBJECT ORIENTED PROGRAMMING

Objectoriented programming. Object-oriented programming was first developed as aconvenient approach to implement simulation problems and distributed operating systems. The notions of objects, classes, and message sending were first introduced in the Simula lan- guage. Organizing code and data around the objects they represent proved to be a very general and natural way to conceptualize applica- tions. Objects can possess local storage in the form of slots, and they have arepertoireof behaviorscalled methods. Anobject’smethod is invoked by sending a message to the object. Most object-oriented languages use three types of objects: classes, metaclasses, and instances. Aclass represents a template for all members of a set of objects. For example, the class Programmer defines aset of objects that represent peoplewho write programs. The class Programmer may have asuper- class, such as Person or Employee, and it can in turn have subclasses such as LispHacker. Aclass object contains adescription of the class’s members. For ex- ample, the class Programmer defines a set of objects that represent people who write programs. It may specify that all Programmer objects should have a slot to store the list of programming languages they use. The class Programmer may have a superclass, such a Person or Em- ployee. It can also have subclasses such as LispHacker. Members of a class are called instances of the class. For example, Trn might be an instance of the class LispHacker. Metaclasses are a special category of class objects whose in- stances are always classes. A major convenience of the object- oriented paradigm is that objects can inherit default characteristics (methodsand slot values) from their superclasses. Theclass Program- mer, for example, could inherit variables such as Employer and Em- ployee Number from the superclass Employee Styles’ strengths. Both styles have their strengths. Object-oriented programming is particularly useful for problems where data objects can be categorized hierarchically. The notions of inheritance and data encapsulation encourage a structured imple- mentation style and enhance program maintainability. Indeed, it has been said that object-oriented programming is to the 1980swhat struc- tured programming was to the 1970s. Object-oriented programming has been used extensively in model building, computer-based simula- tions, and as a tool for knowledge representation in expert systems. Prolog, with as built-in faalities for backtracking and unification, is a natural choice for any application requiring deductive retrieval. Logic programming has been used in many areas, including computational linguistics, database research, and knowledge-based systems.

- Good structure and modularity
- Inheritance
- Communication between processes
- Not appropriate for complex algorithmic problems
