Experiment : 01

Aim :

Introduction to Artificial Intelligence.

Theory :

Artificial Intelligence is an approach to make a computer, a robot, or a product to think about ***"how smart human thinks"***. AI is a study of how the human brain thinks, learn, decide and work when it tries to solve problems. And finally, this study outputs intelligent software systems. The aim of AI is to improve computer functions which are related to human knowledge, for example, reasoning, learning, and problem-solving.

Intelligence is intangible and is composed of:

- Reasoning.
- Learning.
- Problem Solving.
- Perception.
- Linguistic Intelligence.

Importance of AI

- ❖ With the help of AI, one can create such software or devices which can solve real-world problems very easily and with accuracy such as health issues, marketing, traffic issues, etc.
- ❖ With the help of AI, one can create your personal virtual Assistant, such as Cortana, Google Assistant, Siri, etc.
- ❖ With the help of AI, one can build such Robots which can work in an environment where survival of humans can be at risk.

Objectives of AI

- ❖ Replicate human intelligence.
- ❖ Solve Knowledge-intensive tasks.
- ❖ An intelligent connection of perception and action
- ❖ Building a machine which can perform tasks that requires human intelligence like proving a theorem, playing chess, plan some surgical operation, driving a car in traffic, etc.
- ❖ Creating some system which can exhibit intelligent behaviour, learn new things by itself, demonstrate, explain, and can advise to its user.

Languages Used

There are primarily two computer languages used in artificial intelligence work, LISP and PROLOG.

LISP, which is short for List Processing, was created by John McCarthy of Stanford University. It looks klutzy but it is based upon the lambda calculus and works quite well for computation associated with artificial intelligence. PROLOG has an elegant formulation but it does not have the range of application that LISP has. The Japanese when they formulated the Fifth Generation project chose PROLOG over LISP as a programming language. This was perhaps one of the factors that contributed to the failure of the Fifth Generation project. Nevertheless PROLOG is worth knowing for its power in solving questions about relationships.

Introduction to Prolog

Short for Programming Logic, Prolog is a high-level programming language based on formal logic. Unlike traditional programming languages that are based on performing sequences of commands, Prolog is based on defining and then solving logical formulas. Prolog is sometimes called a declarative language or a rule-based language because its programs consist of a list of facts and rules. Prolog is widely used for artificial intelligence applications, particularly expert systems.

Prolog has its roots in first-order logic, a formal logic, and unlike many other programming languages, Prolog is declarative: the program logic is expressed in terms of relations, represented as facts and rules. A computation is initiated by running a query over these relations. The language was first conceived by a group around Alain Colmerauer in Marseille, France, in the early 1970s and the first Prolog system was developed in 1972 by Colmerauer with Philippe Roussel. Prolog is well-suited for specific tasks that benefit from rule-based logical queries such as searching databases, voice control systems, and filling templates.

Introduction to Lisp

Lisp was originally created as a practical mathematical notation for computer programs, influenced by the notation of Alonzo Church's lambda calculus. It quickly became the favoured programming language for artificial intelligence (AI) research. As one of the earliest programming languages, Lisp pioneered many ideas in computer science, including tree data structures, automatic storage management, dynamic typing, conditionals, higher-order functions, recursion, the self-hosting compiler, and the read–eval–print loop. The name LISP derives from "LISt Processor". Linked lists are one of Lisp's major data structures, and Lisp source code is made of lists. Thus, Lisp programs can manipulate source code as a data structure, giving rise to the macro systems that allow programmers to create new syntax or new domain-specific languages embedded in Lisp.