1) ***Define Artificial Intelligence (AI) and provide examples of its applications***.

A. In today's world, technology is growing very fast, and we are getting in touch with different new technologies day by day. One of the booming technologies of computer science is Artificial Intelligence which is ready to create a new revolution in the world by making intelligent machines. The Artificial Intelligence is now all around us. It is currently working with a variety of subfields, ranging from general to specific, such as self-driving cars, playing chess, proving theorems, playing music, Painting e.tc.

AI is one of the fascinating and universal fields of Computer science which has a great scope in future. AI holds a tendency to cause a machine to work as a human.

Artificial Intelligence is composed of two words **Artificial** and **Intelligence**, where Artificial defines *"man-made,"* and intelligence defines *"thinking power"*, hence AI means *"a man-made thinking power."*

Application of AI

Artificial Intelligence has various applications in today's society. It is becoming essential for today's time because it can solve complex problems with an efficient way in multiple industries, such as Healthcare, entertainment, finance, education, etc. AI is making our daily life more comfortable and fast.

Following are some sectors which have the application of Artificial Intelligence:

- **AI in Astronomy:**

Artificial Intelligence can be very useful to solve complex universe problems. AI technology can be helpful for understanding the universe such as how it works, origin, etc.

- **AI in Healthcare:**

In the last, five to ten years, AI becoming more advantageous for the healthcare industry and going to have a significant impact on this industry.

Healthcare Industries are applying AI to make a better and faster diagnosis than humans. AI can help doctors with diagnoses and can inform when patients are worsening so that medical help can reach to the patient before hospitalization.

- **AI in Gaming:**

AI can be used for gaming purpose. The AI machines can play strategic games like chess, where the machine needs to think of a large number of possible places.

- **AI in Finance:**

AI and finance industries are the best matches for each other. The finance industry is implementing automation, chatbot, adaptive intelligence, algorithm trading, and machine learning into financial processes.

- **AI in Data Security:**

The security of data is crucial for every company and cyber-attacks are growing very rapidly in the digital world. AI can be used to make your data more safe and secure. Some examples such as AEG bot, AI2 Platform,are used to determine software bug and cyber-attacks in a better way.

- **AI in Social Media:**

Social Media sites such as Facebook, Twitter, and Snapchat contain billions of user profiles, which need to be stored and managed in a very efficient way. AI can organize and manage massive amounts of data. AI can analyze lots of data to identify the latest trends, hashtag, and requirement of different users.

- **AI in Travel & Transport:**

AI is becoming highly demanding for travel industries. AI is capable of doing various travel related works such as from making travel arrangement to suggesting the hotels, flights, and best routes to the customers. Travel industries are using AI-powered chatbots which can make human-like interaction with customers for better and fast response.

- **AI in Automotive Industry:**

Some Automotive industries are using AI to provide virtual assistant to their user for better performance. Such as Tesla has introduced TeslaBot, an intelligent virtual assistant.Various Industries are currently working for developing self-driven cars which can make your journey more safe and secure.

- **AI in Robotics:**

Artificial Intelligence has a remarkable role in Robotics. Usually, general robots are programmed such that they can perform some repetitive task, but with the help of AI, we can create intelligent robots which can perform tasks with their own experiences without pre-programmed.

Humanoid Robots are best examples for AI in robotics, recently the intelligent Humanoid robot named as Erica and Sophia has been developed which can talk and behave like humans.

- **AI in Agriculture:**

Agriculture is an area which requires various resources, labor, money, and time for best result. Now a day's agriculture is becoming digital, and AI is emerging in this field. Agriculture is applying AI as agriculture robotics, solid and crop monitoring, predictive analysis. AI in agriculture can be very helpful for farmers.

- **AI in education:**

AI can automate grading so that the tutor can have more time to teach. AI chatbot can communicate with students as a teaching assistant.

AI in the future can be work as a personal virtual tutor for students, which will be accessible easily at any time and any place.

2) Differentiate between supervised and unsupervised learning techniques in ML.

A. Machine learning is already an important part of how modern organisation and services function. Whether in social media platforms, healthcare, or finance, machine learning models are deployed in a variety of settings. But the steps needed to train and deploy a model will differ depending on the task at hand and the data that’s available.

The main difference between supervised vs unsupervised learning is the need for labelled training data. Supervised machine learning relies on labelled input and output training data, whereas unsupervised learning processes unlabelled or raw data. In supervised machine learning the model

learns the relationship between the labelled input and output data. Models are finetuned until they can accurately predict the outcomes of unseen data. However, labelled training data will often be resource intensive to create. Unsupervised machine learning on the other hand learns from unlabelled raw training data. An unsupervised model will learn relationships and patterns within this unlabelled dataset, so is often used to discover inherent trends in a given dataset.

So overall, supervised and unsupervised machine learning are different in the approach to training and the data the model learns from. But as a result, they also differ in their final application and specific strengths. Supervised machine learning models are generally used to predict outcomes for unseen data. This could be predicting fluctuations in house prices or understanding the sentiment of a message.

Models are also used to classify unseen data against learned patterns. On the other hand, unsupervised machine learning techniques are generally used to understand patterns and trends within unlabelled data. This could be clustering data due to similarities or differences, or identifying underlying patterns within datasets. Unsupervised machine learning can be used to cluster customer data in marketing campaigns, or to detect anomalies and outliers.

The main differences of supervised vs unsupervised learning include:

- The need for labelled data in supervised machine learning.
- The problem the model is deployed to solve. Supervised machine learning is generally used to classify data or make predictions, whereas unsupervised learning is generally used to understand relationships within datasets.
- Supervised machine learning is much more resource-intensive because of the need for labelled data.

In unsupervised machine learning it can be more difficult to reach adequate levels of explainability because of less human oversight.

3) What is Python? Discuss its main features and advantages.

A. Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

Advantages of Python:

- Easy to Learn and Use.
- Free and Open-Source
- Rapid Development
- Interpreted Language
- Wide Range of Libraries and Frameworks
- Dynamically Typed
- Portability
- Strong Community Support
- Cross-Platform Compatibility
- Strong Community Support
- Integration and Extensibility
- Scalability and Performance
- Versatility and Flexibility

some of the important features of Python that make it widely popular are:

1. *Easy to Learn:*

   This is one of the most important reasons for the popularity of Python. Python has a limited set of keywords. Its features such as simple syntax, usage of indentation to avoid clutter of curly brackets and dynamic typing that doesn't necessitate prior declaration of variable help a beginner to learn Python quickly and easily.

2. Interpreter Based:

Instructions in any programming languages must be translated into machine code for the processor to execute them. Programming languages are either compiler based or interpreter based.

In case of a compiler, a machine language version of the entire source program is generated. The conversion fails even if there is a single erroneous statement. Hence, the development process is tedious for the beginners. The C family languages (including C, C++, Java, C# etc) are compiler based.

Python is an interpreter based language. The interpreter takes one instruction from the source code at a time, translates it into machine code and executes it. Instructions before the first occurrence of error are executed. With this feature, it is easier to debug the program and thus proves useful for the beginner level programmer to gain confidence gradually. Python therefore is a beginner-friendly language.

3. Interactive:

Standard Python distribution comes with an interactive shell that works on the principle of REPL (Read – Evaluate – Print – Loop). The shell presents a Python prompt >>>. You can type any valid Python expression and press Enter. Python interpreter immediately returns the response and the prompt comes back to read the next expression.

The interactive mode is especially useful to get familiar with a library and test out its functionality. You can try out small code snippets in interactive mode before writing a program.

4. MultiParadigm:

Python is a completely object-oriented language. Everything in a Python program is an object. However, Python conveniently encapsulates its object orientation to be used as an imperative or procedural language – such as C. Python also provides certain functionality that resembles functional programming. Moreover, certain third-party tools have been developed to support other programming paradigms such as aspect-oriented and logic programming.

5. Standard Library:

Even though it has a very few keywords (only Thirty Five), Python software is distributed with a standard library made of large number of modules and packages. Thus Python has out of box support for programming needs such as serialization, data compression, internet data handling, and many more. Python is known for its batteries included approach.

6.Open Source and Cross Platform:

Python's standard distribution can be downloaded from https://www.python.org/downloads/ without any restrictions. You can download pre-compiled binaries for various operating system platforms. In addition, the source code is also freely available, which is why it comes under open source category.

7. GUI Applications:

Python's standard distribution has an excellent graphics library called TKinter. It is a Python port for the vastly popular GUI toolkit called TCL/Tk. You can build attractive user-friendly GUI applications in Python. GUI toolkits are generally written in C/C++. Many of them have been ported to Python. Examples are PyQt, WxWidgets, PySimpleGUI etc.

8. Database Connectivity:

Almost any type of database can be used as a backend with the Python application. DB-API is a set of specifications for database driver software to let Python communicate with a relational database. With many third party libraries, Python can also work with NoSQL databases such as MongoDB.

9. Extensible:

The term extensibility implies the ability to add new features or modify existing features. As stated earlier, CPython (which is Python's reference implementation) is written in C. Hence one can easily write modules/libraries in C and incorporate them in the standard library. There are other implementations of Python such as Jython (written in Java) and IPython (written in C#). Hence, it is possible to write and merge new functionality in these implementations with Java and C# respectively.

10. Active Developer Community:

As a result of Python's popularity and open-source nature, a large number of Python developers often interact with online forums and conferences. Python Software Foundation also has a significant member base, involved in the organization's mission to "Promote, Protect, and Advance the Python Programming Language"

Python also enjoys a significant institutional support. Major IT companies Google, Microsoft, and Meta contribute immensely by preparing documentation and other resources.

4) *What are the advantages of using Python as a programming language for AI and ML?*

A. Artificial intelligence and machine learning are evolving with each passing day. The different solutions play an important role in easing the entire programming solution.

While numerous programming languages can be helpful, leaders across the industry rely on Python itself.

Advantages of using python as a programming language for AI and ML are:

1. Simple and consistent:

Python provides the benefit of a reasonable code. AI and ML require solving complex algorithms. However, the simplicity of Python will ensure that developers can easily write the codes. Whenever opting for Python development, businesses should consider understanding the code.

2. Better library ecosystem:

Comparatively, Python has a better library system that is crucial for the development process. A library refers to a group of modules with a pre-written code set. Depending on these codes, the users can focus on promoting functionality.

3. Flexible:

One of the main reasons companies hire expert python developers is that they offer flexibility. The programming language provides the benefits of choosing between OOPS and scripting. Furthermore, you can consider recompiling the source code to bring any changes.

Python as a flexible platform allows developers to choose from different programming styles. As a result, depending on the developer's needs, they can also consider combining various styles.

4. Better visualization option

We already know that Python has a variety of libraries online, and most of these libraries also feature exclusive visualization tools. When it comes to artificial intelligence, the developers need to highlight the visuals accurately to get attention. Furthermore, it also plays a vital role in presenting the data in a readable format.

5. Readability:

Python is one of the leading platforms that bring the benefit of readability. Since it is an easy-to-read programming language, beginners can easily share and change the codes. Unlike the other programming languages, Python is not at all complicated.

The ease of use of the programming language plays an important role in exchanging ideas, tools, and algorithms. Therefore, Ai-professionals can easily use Python to bring minor or major changes into their projects.

Apart from the general schedule, numerous tools help create an interactive design. These external tools help to add extra features such as tab completion, debugging, testing, and more. As a result, it plays an essential role in facilitating the work schedule.

6. Platform independence:

Not all programming languages are platform-independent. However, Python, a versatile programming language, does benefit platform independence. Therefore, Python can easily function on different platforms such as Unix, Linux, macOS, Windows, and more.

Developers across different platforms can collaborate to build the right platform. As a result, developers who need to bring small changes into the design can execute codes within the program.

Since numerous packages are available online, developers can choose what suits them the most. Developers will have the flexibility to use different packages to help them execute or run their codes across different platforms.

Platform independence, in the case of using Python for AI & ML development, refers to the ideal of allowing the developers to implement all attributes upon one machine. Following that, those attributes can again be used over another machine without the need for making any changes.

The best thing about Python is its independent language. It is supported by various platforms, including macOS, Windows, Linux, and others.

Python code is used as a standalone program for most of the common OSs. Thus, it means that Python language can be distributed without the need for a Python interpreter. The platform independence feature of Python plays an important role in saving time and money. Therefore, it helps to make the entire development process fast and easy.

7. Rapid development:

One of the main reasons developers love Python is that it benefits fast prototyping. As long as you are familiar with the stacks, the developers can prevent wasting time. Therefore, the developers will not need to waste time and learn AI development easily.

Many developers consider Python to be as simple as English as far as writing and readability are concerned. As a Python development professional, you will not need to learn complicated codes. Thanks to the availability of numerous libraries, AI and ML development becomes easy, thereby simplifying the project.

9. Less coding:

Artificial intelligence is rapidly developing. Therefore, the application of AI will require you to indulge in using numerous algorithms. When you use Python for AI and ML development, you will access numerous predefined packages.

Therefore, Python does not require you to indulge in hard-core coding because you already have predefined packages. The programming language also plays an important role in making the entire process easy by bringing in 'check your code.' As a result, you don't have to rely on testing the code because the platform will do it for you.

9. Speed of execution

Since Python is a readable programming language, the formulas can be executed quickly. Machine Learning, especially Deep learning, requires long training sessions, and these training sessions can last for days even. Nonetheless, Python has a faster execution speed which is crucial.

The syntax of Python is quite easy to understand and is very developer-friendly. As there is an abundance of libraries and frameworks, software development can be executed with high proficiency and efficiency on priority. By using Python, you can do a lot with just a few of the code lines. You can use it for developing prototypes and enhancing productivity.

Moreover, there are a lot of code testing and reviewing tools available with Python. Hence, the developers have the flexibility of quickly checking the quality and correctness of the code.

5) ***Discuss the importance of indentation in Python code.***
A. In Python, indentation refers to the spacing at the beginning of a line of code that determines its grouping and hierarchy within the program's structure. Unlike many programming languages that use braces ({}) or other explicit symbols to denote code blocks, Python uses indentation to signify the beginning and end of blocks of code.
The primary purpose of indentation in Python is to define the scope of statements, such as those within loops, conditionals, functions, and classes. Consistent and proper indentation is crucial for the interpreter to understand the logical structure of the code. Indentation is not just a matter of style or convention in Python.
Rules of Indentation in Python

- Python's default indentation spaces are four spaces. The number of spaces, however, is entirely up to the user. However, a minimum of one space is required to indent a statement.

- Indentation is not permitted on the first line of Python code.
- Python requires indentation to define statement blocks.
- A block of code must have a consistent number of spaces.
- To indent in Python, whitespaces are preferred over tabs. Also, use either whitespace or tabs to indent; mixing tabs and whitespaces in indentation can result in incorrect indentation errors

6) Discuss Define a variable in Python. Provide examples of valid variable names.

A. A Python variable is a reserved memory location to store values. In other words, a variable in a python program gives data to the computer for processing. Every value in Python has a datatype. Different data types in Python are Numbers, List, Tuple, Strings, Dictionary, etc. Variables can be declared by any name or even alphabets like a, aa, abc, etc

Variable Naming Rules in Python

1.Variable name should start with letter(a-zA-Z) or underscore (_).

Valid : age , _age , Age Invalid : 1age
2.In variable name, no special characters allowed other than underscore (_).
Valid : age_ , _age Invalid : age_*
3.Variables are case sensitive. age and Age are different, since variable names are case sensitive.
4.Variable name can have numbers but not at the beginning.
Example: Age1
5.Variable name should not be a Python keyword.Keywords are also called as reserved words.
Example pass, break, continue.. etc are reserved for special meaning in Python. So, we should not declare keyword as a variable name. How to Declare and use a Variable Let see an example. We will declare variable "a" and print it. a=100 print (a)

7) Explain the difference between a keyword and an identifier in Python
A.

Keyword	**Identifier**
Keywords are the reserved words with a special meaning.	Identifiers are the user-defined names of variables, functions, etc.
They are written in lower case except for True, False, and None.	Need not be written in lowercase.
It helps to identify a specific property that exists within Python.	It identifies the name of the particular entity.
Contains only letters	Contains letters, underscore, and digits.

Keyword	Identifier
Example:- or, raise, pass	Example:- maxCount, minNum1, etc

8*) List the basic data types available in Python.*

A. In computer programming, data types specify the type of data that can be stored inside a variable. For example,

num=24

Here, 24 (an integer) is assigned to the num variable. So the data type of num is of the int class.

Python Numeric Data type

In Python, numeric data type is used to hold numeric values.

Integers, floating-point numbers and complex numbers fall under Python numbers category. They are defined as `int`, `float` and `complex` classes in Python.

`int` - holds signed integers of non-limited length.

`float` - holds floating decimal points and it's accurate up to **15** decimal places.

`complex` - holds complex numbers.

Python List Data Type

List is an ordered collection of similar or different types of items separated by commas and enclosed within brackets `[ ]`. For example,

Languages=[“swift”, “java”,”python”]

Python Tuple Data Type

Tuple is an ordered sequence of items same as a list. The only difference is that tuples are immutable. Tuples once created cannot be modified.

In Python, we use the parentheses `( )` to store items of a tuple. For example

Product=(‘xbox’,499.99)

Python String Data Type

String is a sequence of characters represented by either single or double quotes.

For example,

Name=’python’

Print(name)

Python Set Data Type

Set is an unordered collection of unique items. Set is defined by values separated by commas inside braces { }.

For example,

 # create a set named student_id

student_id = {112, 114, 116, 118, 115}

 # display student_id elements

print(student_id)

 # display type of student_id

print(type(student_id))

Output

{112, 114, 115, 116, 118}

<class 'set'>

Python Dictionary Data Type

Python dictionary is an ordered collection of items. It stores elements in key/value pairs.

Here, keys are unique identifiers that are associated with each value.

Let's see an example,

 # create a dictionary named capital_city

capital_city = {'Nepal': 'Kathmandu', 'Italy': 'Rome', 'England': 'London'}

print(capital_city)

Run Code

Output

{'Nepal': 'Kathmandu', 'Italy': 'Rome', 'England': 'London'}

9) Describe the syntax for an if statement in Python.

A. In Python the **if statement** is used for conditional execution or branching. An if statement is one of the **control structures**. (*A control structure controls the flow of the program.*)

The if statement may be combined with certain operator such as equality (==), greater than (>=), smaller than (<=) and not equal (!=). Conditions may be combined using the keywords **or** and **and**.

In the example below we show the use *if* statement, a control structure. An if statement evaluates data (a condition) and makes a choice.

Lets have al look at a basic if statement. In its basic form it looks like this:

if <condition>:

<statement>

in this form

is the condition evaluated as a Boolean, it can either be True or False.

is one more lines of code. Each of those lines must indented with four spaces.

Example:

```
>>> x = 3
>>> if x < 10:
...     print('x below ten')
...
x below ten
>>> if x > 10:
...     print('x is greater than ten')
...
>>> if x > 1 and x < 4:
...     print('x is in range')
...
x is in range
>>>
```

It's very important to have four spaces for the statements. Every if statement needs a colon. More than one condition can be combined using the *and* keyword.

10) Explain the purpose of the elif statement in Python.

A. 'Elif' stands for 'else if' and is used in Python programming to test multiple conditions. It is written following an if statement in Python to check an alternative condition if the first condition is false. The code block under the elif statement will be executed only if its condition is true.

Syntax:

```
if condition1:
    statement to execute if condition1 is true
elif condition2:
    statement to execute if condition2 is true
else:
    statement to execute if both conditions are false
```

Here,

- If condition 1 is true, the statement present inside the if block will be executed.
- If condition 1 is false, then the elif condition will be checked.
- If the elif condition is true, the statement present inside the elif block will be executed.
- If both conditions are false, the statement present inside the else block will be executed.

Multiple elif statements can be used in Python by nesting them inside one another.

For example:

if condition1:

statement to execute if condition1 is true

elif condition2:

statement to execute if condition2 is true

elif condition3:

statement to execute if condition3 is true

else:

statement to execute if all conditions are false

Here,

- If condition 1 is true, the statement present inside the if block will be executed. If condition 1 is false, then the elif condition 2 will be checked.
- If condition 2 is true, the statement present inside the elif block will be executed.
- Similarly, if condition 3 is true, the statement present inside the elif block will be executed.
- If all conditions are false, the statement present inside the else block will be executed.