COURSE OUTCOME 1

DATE: 26/09/2024

1. Familiarizing Integrated Development Environment (IDE), CodeAnalysis Tools

An integrated development environment (IDE) refers to a software application that offers computer programmers with extensive software development abilities. IDEs most often consist of a source code editor, build automation tools, and a debugger. Most modern IDEs have intelligent code completion. An IDE enables programmers to combine the different aspects of writing a computer program and increase programmer productivity by introducing features like editing source code, building executable, and debugging. IDEs are usually more feature-rich and include tools for debugging, building and deploying code. An IDE typically includes:

- A source code editor
- A compiler or interpreter
- An integrated debugger
- A graphical user interface (GUI)

A code editor is a text editor program designed specifically for editing source code. It typically includes features that help in code development, such as syntax highlighting, code completion, and debugging. The main difference between an IDE and a code editor is that an IDE has a graphical user interface (GUI) while a code editor does not. An IDE also has features such as code completion, syntax highlighting, and debugging, which are not found in a code editor. Code editors are generally simpler than IDEs, as they do not include many other IDE components. As such, code editors are typically used by experienced developers who prefer to configure their development environment manually. Some IDEs are given below:

1. IDLE

IDLE (Integrated Development and Learning Environment) is a default editor that accompanies Python. This IDE is suitable for beginner-level developers. The IDLE tool can be used on Mac OS, Windows, and Linux. The most notable features of IDLE include:

- Ability to search for multiple files
- Interactive interpreter with syntax highlighting, and error and i/o messages
- Smart indenting, along with basic text editor features
- A very capable debugger
- A great Python IDE for Windows

2. **PyCharm**

PyCharm is a widely used Python IDE created by JetBrains This IDE is suitable for professional developers and facilitates the development of large Python projects

The most notable features of PyCharm include:

- Support for JavaScript, CSS, and TypeScript
- Smart code navigation
- Quick and safe code refactoring
- Support features like accessing databases directly from the IDE

3. **Visual Studio Code**

Visual Studio Code (VS Code) is an open-source (and free) IDE created by Microsoft. It finds great use in Python development. VS Code is lightweight and comes with powerful features that only some of the paid IDEs offer. The most notable features of Visual Studio Code include Git integration and Code debugging within the editor.

4. Sublime Text 3

Sublime Text is a very popular code editor. It supports many languages, including Python. It is highly customizable and also offers fast development speeds and reliability. The most notable features of Sublime Text 3 include:

- Syntax highlighting
- Custom user commands for using the IDE
- Efficient project directory management
- It supports additional packages for the web and scientific Python development

5. Atom

Atom is an open-source code editor by GitHub and supports Python development. Atom is similar to Sublime Text and provides almost the same features emphasis on speed and usability. The most notable features of Atom include:

- Support for a large number of plugins
- Smart autocompletion
- Supports custom commands for the user to interact with the editor
- Support for cross-platform development

5. Jupyter

Jupyter is widely used in the field of data science. It is easy to use, interactive and allows live code sharing and visualization. The most notable features of Jupyter include:

- Supports for the numerical calculations and machine learning workflow
- Combine code, text, and images for greater user experience
- Intergeneration of data science libraries like NumPy, Pandas, and Matplotlib

6. Spyder

Spyder is an open-source IDE most commonly used for scientific development. Spyder comes with Anaconda distribution, which is popular for data science and machine learning. The most notable features of Spyder include:

- Support for automatic code completion and splitting
- Supports plotting different types of charts and data manipulation
- Integration of data science libraries like NumPy, Pandas, and Matplotlib

Code Analysis Tools

Source code analysis tools, also known as Static Application Security Testing (SAST) Tools, can help analyse source code or compiled versions of code to help find security flaws. SAST tools can be added into IDE. Such tools can help to detect issues during software development. Static code analysis techniques are used to identify potential problems in code before it is deployed, allowing developers to make changes and improve the quality of the software. Three techniques include syntax analysis, data and control flow analysis, and security analysis.

SonarQube (Community Edition) is an open source static + dynamic code analysis platform developed by SonarSource for continuous inspection of code quality to perform fully automated code reviews / analysis to detect code smells, bugs, performance enhancements and security vulnerabilities.

DATE: 8-10-2024

2. Display future leap years from current year to a final year entered by user?

PROGRAM

```
year=int(input("Enter the current year:"))
fyear=int(input("Enter the final year:"))
while(year<fyear):
    if(year%4==0) and (year%100!=0) or (year%400==0):
        print(year," is leap year")
    year=year+1
```

OUTPUT

```
Enter the current year:2020
Enter the final year:2050
2020  is leap year
2024  is leap year
2028  is leap year
2032  is leap year
2036  is leap year
2040  is leap year
2044  is leap year
2048  is leap year
```

OUTPUT

Enter the current year:2024

Enter the final year:2060

2024 is leap year

2028 is leap year

2032 is leap year

2036 is leap year

2040 is leap year

2044 is leap year

2048 is leap year

2052 is leap year

2056 is leap year

DATE: 10-10-2024

3. List comprehensions:

   (a) Generate positive list of numbers from a given list of integers

   (b) Square of N numbers

   (c) Form a list of vowels selected from a given word

   (d) List ordinal value of each element of a word (Hint: use ord() to get ordinal values)

PROGRAM

```
l=input("Enter the element in seperatly")
li=[int(num)for num in l.split()]
pl=[num for num in li if num >0]
print("List of Positive number are",pl)

l=input("Enter the numbers:")
li=[int(num)for num in l.split()]
sl=[num*num for num in li]
print(sl)

b=input("Enter a word:")
vowel=[char for char in b if char.lower() in "aeiou"]
print("vowels are:",vowel)

word = input("Enter a word:")
ordinal_values = [ord(char) for char in word]
 print(ordinal_values)
```

OUTPUT

Enter the element in seperatly:8 -5 2 9

List of Positive number are [8, 2, 9]

Enter the numbers:5 8 6 8 5

[25, 64, 36, 64, 25]

Enter a word:hello

vowels are: ['e', 'o']

Enter a word: python [112, 121, 116, 104, 111, 110]

OUTPUT

Enter the element in seperatly 2 3 8 -1

List of Positive number are [2, 3, 8]

Enter the numbers:5 9 7 10

[25, 81, 49, 100]

Enter a word:python program

vowels are: ['o', 'o', 'a']

Enter a word: programming

[112, 114, 111, 103, 114, 97, 109, 109, 105, 110, 103]

DATE: 8-10-2024

4.Count the occurrences of each word in a line of text.?

PROGRAM

```
line_of_text = input("Enter a line of text: ")
words = line_of_text.split()
word_count = {}
for word in words:
    if word in word_count:
        word_count[word] += 1
    else:
        word_count[word] = 1
print(word_count)
```

OUTPUT

```
Enter a line of text: hello hai hello python program
{'hello': 2, 'hai': 1, 'python': 1, 'program': 1}
```

OUTPUT

```
Enter a line of text: python is an high level language
{'python': 1, 'is': 1, 'an': 1, 'high': 1, 'level': 1, 'language': 1}
```

DATE: 15-10-2024

5.Prompt the user for a list of integers. For all values greater than 100, store ‘over’ instead. using comprehension method?

PROGRAM

```
x=input("enter the integers: ")
result=['over' if int(num)>100 else num  for num in x.split() ]
print(result)
```

OUTPUT

```
enter the integers: 100 50 150 200
['100', '50', 'over', 'over']
```

OUTPUT

```
enter the integers: 300 500 60 50
['over', 'over', '60', '50']
```

DATE: 24-10-2024

6. Store a list of first names. Count the occurrences of ‘a’ within the list?

PROGRAM

```
list=["anna","apple","amar","rafeil"]
count_a=0
for X in list:
    count_a += X.count('a')
print(list)
print(f"the letter 'a' occur {count_a} times in the list")
```

OUTPUT

['Alice', 'Bob', 'Anna', 'Charlie', 'David', 'Eva', 'Michael', 'Sara']

The letter 'a' occurs 9 times in the list.

DATE:28-10-2024

7. Enter 2 lists of integers. Check

(a) Whether list are of same length

(b) whether list sums to same value

(c) whether any value occur in both ?

PROGRAM

```
a = [int(i) for i in input("Enter the first list of integers: ").split()]
b = [int(i) for i in input("Enter the second list of integers: ").split()]
same = len(a) == len(b)
sum1 = sum(a) == sum(b)
common_values = [i for i in a if i in b]
print(f"Are the lists of the same length: {'Yes' if same else 'No'}")
print(f"Do the lists sum to the same value: {'Yes' if sum1 else 'No'}")
print(f"Common values in both lists: {common_values if common_values
else 'None'}")
```

OUTPUT

```
Enter the first list of integers: 1 2 3 5 6
Enter the second list of integers: 2 5 6 9 8
Are the lists of the same length: Yes
Do the lists sum to the same value: No
Common values in both lists: [2, 5, 6]
```

OUTPUT

Enter the first list of integers:1 34 5 21

Enter the second list of integers:12 3 5 21

Are the lists of the same length Yes

Do the lists sum to the same value No

Common values in both lists: [5, 21]

DATE: 24-10-2024

8. Get a string from an input string where all occurrences of first character replaced with ‘$’, except first character

.[eg: onion -> oni$n]

PROGRAM

```
string = input("Enter a string: ")
if string:
    first_char =string[0]
    modified_string = first_char + string[1:].replace(first_char, '$')
else:
    modified_string = ""
print(f"Modified string: {modified_string}")
```

OUTPUT

Enter a string: onion

Modified string: oni$n

OUTPUT

Enter a string: river

Modified string: rive$

DATE:28-10-2024

9. Create a string from given string where first and last characters exchanged. [eg: python -> nythop]

PROGRAM

```
s = input("Enter the string:")
result = s[-1] + s[1:-1] + s[0]
print(result)
```

OUTPUT

```
Enter the string: python
nythop
```

OUTPUT

```
Enter the string:program
mrograp
```

DATE:2-10-2024

10.write a program to input radius of circle and find the area of circle ?

PROGRAM

```
r=int(input("Enter radius of circle:"))
pi=3.14
area=pi*r*r
print("Area of cirlce :",area)
```

OUTPUT

```
Enter radius of circle: 5
Area of cirlce : 78.5
```

OUTPUT

```
Enter radius of circle:6
Area of cirlce : 113.03999999999999
```

DATE:2-10-2024

11. Find biggest of 3 numbers entered.?

PROGRAM

```
a=int(input("Enter first value:"))
b=int(input("Enter second value:"))
c=int(input("Enter third value:"))
if a>c and a>b:
   print(a,"is greater")
elif b>c:
   print(b,"is greater")
else:
   print(c,"is largest")
```

OUTPUT

```
Enter first value:10
Enter second value:5
Enter third value:6
10 is greater
```

OUTPUT

```
Enter first value:10
Enter second value:5
Enter third value:60
60 is largest
```

DATE:8-10-2024

12. Accept a file name from user and print extension of that.

PROGRAM

```
filename = input("Please enter a file name: ")
dot_index = filename.rfind('.')

if dot_index != -1 and dot_index != len(filename) - 1:
    extension = filename[dot_index + 1:]
    print(f"The file extension is: {extension}")
else:
    print("No valid file extension found.")
```

OUTPUT

```
Please enter a file name: file.py
The file extension is: py
```

OUTPUT

```
Please enter a file name: sum.py
The file extension is: py
```

DATE:2-10-2024

13.Create a list of colors from comma-separated color names entered by user. Display first and last colors

PROGRAM

```
colors_input = input("Please enter a list of colors (comma-separated): ")
colors = [color for color in colors_input.split(',')]
print(colors)


if colors:
   print(f"The first color is: {colors[0]}")
    print(f"The last color is: {colors[-1]}")
else:
    print("No colors were entered.")
```

OUTPUT

```
Please enter a list of colors (comma-separated): black,blue,red,green
['black', 'blue', 'red', 'green']
The first color is: black
The last color is: green
```

OUTPUT

```
Please enter a list of colors (comma-separated): red,yellow,pink,blue
['red', 'yellow', 'pink', 'blue']
The first color is: red
The last color is: blue
```

DATE:9-10-2024

14. Accept an integer n and compute n+nn+nnn?

PROGRAM

```
n=int(input("Enter a value:"))
n_str = str(n)
nn = int(n_str * 2)
nnn = int(n_str * 3)
result = n + nn + nnn
print(result)
```

OUTPUT

```
Enter a value:5
615
```

OUTPUT

```
Enter a value:6
738
```

DATE:15-10-2024

15.Print out all colors from color-list1 not contained in color-list2.create color list given as user input. use list comprehension?

PROGRAM

```
list1=[i for i in input("enter first list of colors: ").split()]
list2=[i for i in input("enter the second list of colors: ").split()]
c=[i for i in list1 if i not in list2 ]
print("colors in list-1 not in list-2= ",c)
```

OUTPUT

```
enter first list of colors: green blue black
enter the second list of colors: orange blue black
colors in list-1 not in list-2=  ['green']
```

OUTPUT

```
enter first list of colors: black,red,white,pink
enter the second list of colors: black,pink,yellow,blue
colors in list-1 not in list-2  ['black,red,white,pink']
```

DATE:16-10-2024

16.Create a single string separated with space from two strings by swapping the character at position 1.

PROGRAM

```
string1 = input("Enter the first string: ")
string2 = input("Enter the second string: ")

if len(string1) > 1 and len(string2) > 1:
    new_string1 = string1[0] + string2[1] + string1[2:]
    new_string2 = string2[0] + string1[1] + string2[2:]
    result = new_string1 + " " + new_string2
    print("Resulting string:", result)
else:
    print("Both strings must have at least 2 characters.")
```

OUTPUT

```
Enter the first string: hello
Enter the second string: python
Resulting string: hyllo pethon
```

OUTPUT

```
Enter the first string: program
Enter the second string: level
Resulting string: peogram lrvel
```

DATE:10-10-2024

17.Sort dictionary in ascending and descending order.

PROGRAM

```
dis={
   "onion":23,
   "tomato":2,
   "chilli":45,
   "beans":32
}
print(dis)
print("Assending order:")
res=dict(sorted(dis.items()))
print(res)
print("Decending order:")
res=dict(sorted(dis.items(),reverse=True))
print(res)
```

OUTPUT

```
{'onion': 23, 'tomato': 2, 'chilli': 45, 'beans': 32}
Assending order:
{'beans': 32, 'chilli': 45, 'onion': 23, 'tomato': 2}
Decending order:
{'tomato': 2, 'onion': 23, 'chilli': 45, 'beans': 32}
```

DATE:18-10-2024

18.Merge two dictionaries.

PROGRAM

```
vegtable={
   1:"beans",
   2:"carrot",
   3:"tomato"
}
fruit={
   1:"orange",
   2:"jackfruit",
   3:"apple",
}
vegtable.update(fruit)
print(vegtable)
print(vegtable|fruit)
```

OUTPUT

```
{1: 'orange', 2: 'jackfruit', 3: 'apple'}
{1: 'orange', 2: 'jackfruit', 3: 'apple'}
```

DATE:15-10-2024

19.Find gcd of 2 numbers?

PROGRAM

```
import math
x=int(input("Enter first numbers:"))
y=int(input("Enter second numbers:"))
print("gcd of this two number:",math.gcd(x,y))
```

OUTPUT

```
Enter first numbers:8
Enter second numbers:6
gcd of this two number: 2
```

OUTPUT

```
Enter first numbers:5
Enter second numbers:10
gcd of this two number: 5
```

DATE:29-10-2024

20.From a list of integers, create a list removing even numbers.?

PROGRAM

```
x=[int(i) for i in input("Enter the integers:").split()]
newlist=[i for i in x if i%2!=0]
print(newlist)
```

OUTPUT - 1

```
Enter the integers:1 2 4 5 6 9
[1, 5, 9]
```

OUTPUT - 2

```
Enter the integers:2 10 50 85 95
[85, 95]
```