COURSE OUTCOME 1

DATE: 26/09/2024

1. Familiarizing Integrated Development Environment (IDE), Code Analysis 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

6. 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

7. 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: 08/10/2024

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

PROGRAM

```
final=int(input("Enter final year:"))
c=2024
for i in range(c,final+1):
        if i%400==0 or i%100!=0 and i%4==0:
                print(i,"is a leap year\n")
```

OUTPUT

Enter final year:2040

2024 is a leap year

2028 is a leap year

2032 is a leap year

2036 is a leap year

2040 is a leap year

Enter final year:2028

2024 is a leap year

2028 is a 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

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

```
l=input("Enter the list of integers seperated by spaces:")
li=[int(num) for num in l.split()]
pl=[num for num in li if num>0]
print("The list of positive numbers are",pl)
```

b) Square of N numbers

```
l=input("Enter the list of integers seperated by spaces:")
li=[int(num) for num in l.split()]
sq=[num*num for num in li if num>0]
print("Square of entered numbers:",sq)
```

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

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

(d) List ordinal value of each element of a word

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

OUTPUT

(a) Enter the list of integers seperated by spaces: 9 67 55 78 78 -56

The list of positive numbers are [9, 67, 55, 78, 78]

(b) Enter the list of integers seperated by spaces:2 6 4 7
Square of entered numbers: [4, 36, 16, 49]
(c) Enter a word: code
vowels are: [‘o’,’e’]

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

(a) Enter the list of integers seperated by spaces: 8 3 -2 -1 63
The list of positive numbers are [8, 3, 63]

(b) Enter the list of integers seperated by spaces:9 2 4
Square of entered numbers: [81, 4, 16]

(c) Enter a word: World
vowels are: [‘o’]

(d) 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:
    word = word.strip('.,!?";:')
    if word in word_count:
        word_count[word] += 1
    else:
        word_count[word] = 1
print(word_count)
```

OUTPUT

```
Enter a line of text: Hello world! Hello Python
{'Hello': 2, 'world': 1, 'Python': 1}


Enter a line of text: Python is fun, and learning Python is even better!
{'Python': 2, 'is': 2, 'fun': 1, 'and': 1, 'learning': 1, 'even': 1, 'better': 1}
```

DATE:15/10/24

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

PROGRAM

```
s=input("Enter list of integers seperated by spaces:")
number=s.split()
result=[]
for n in number:
        num=int(n)
        if num>100:
                result.append("over")
        else:
                result.append(num)
print("New list",result)
```

OUTPUT

```
Enter list of integers seperated by spaces:156 22 33
New list ['over', 22, 33]


Enter list of integers seperated by spaces:23 67 456 54
New list [23,67,'over', 54]
```

DATE: 24/10/2024

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

PROGRAM

```
first_names = ["Fina", "Anna", "Joanna", "Aiyra","Hezzah", "Agnes", " Canny"]
count_a = sum(name.lower().count('a') for name in first_names)
print(first_names)
print(f"The letter 'a' occurs {count_a} times in the list.")
```

OUTPUT

```
["Fina", "Anna", "Joanna", "Aiyra","Hezzah", "Agnes", " Canny"]
The letter 'a' occurs 12 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

```
x=[int(i) for i in input("Enter the first list of integers:").split()]
y=[int(i) for i in input("Enter the second list of integers:").split()]
same=len(x)==len(y)
sum1=sum(x)==sum(y)
common_values = [i for i in x if i in y]
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 integers:9 5 2 1
Enter the second list integers:9 4 2 6
Are the lists of the same length Yes
Do the lists sum to the same value No
Common values in both lists: [9, 2]

Enter the first list of integers:32 5 12 8 4
Enter the second list of integers: 89 7 5 11 4
Are the lists of the same length Yes
Do the lists sum to the same value No
Common values in both lists: [5, 4]
```

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

Enter a string: madam
Modified string: mada$
```

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: coding
godinc

DATE: 08/10/2024

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

PROGRAM

```
r=float(input("Enter Raduis: "))
area=3.14*r*r
print("Area of circle=",area ,"sq.cm")
```

OUTPUT

```
Enter Raduis: 5
Area of circle= 78.5 sq.cm


Enter Raduis: 7
Area of circle= 153.9 sq.cm
```

DATE: 08/10/2024

11.write a program to input 3 integer number and find the larger among the two

PROGRAM

```
x=int(input("Enter first number:"))
y=int(input("Enter second number:"))
z=int(input("Enter third number:"))
if x==y==z:
        print("All numbers are equal")
elif x>y and x>z:
        print(x,"is greater")
elif y>x and y>z:
        print(y, "is greater")
else:
        print(z, "is greater")
```

OUTPUT

```
Enter second number:43
Enter third number:22
Enter first number:3
43 is greater


Enter first number:6
Enter second number:198
Enter third number:64
198 is greater
```

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: hello.py
The file extension is: py


Please enter a file name: function.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.strip() for color in colors_input.split(',')]

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): blue, red, orange, green, pink, teal
The first color is: blue
The last color is: teal

Please enter a list of colors (comma-separated): black, blue, yellow, black
The first color is: black
The last color is: black
```

DATE: 08/10/2024

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

PROGRAM

```
n=int(input("Enter an Integer:"))
s=n+n*11+n*111
print(s)
```

OUTPUT

```
Enter an Integer:6
738

Enter an Integer:5
615
```

DATE:15/10/24

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

PROGRAM

```
l1=[i for i in input("Enter the colors in list 1:").split()]
l2=[i for i in input("Enter the colors in list 2:").split()]
result=[i for i in l1 if i not in l2]
print("Colors in list 1 not in list 2:",result)
```

OUTPUT

```
Enter the colors in list 1: red yellow indigo pink
Enter the colors in list 2: green blue red purple
Colors in list 1 not in list 2 :['yellow', 'indigo', 'pink']


Enter the colors in list 1: red yellow indigo pink
Enter the colors in list 2: indigo red pink green
Colors in list 1 not in list 2 :['yellow']
```

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: cat
Enter the second string: dog
Resulting string: cot dag


Enter the first string: hello
Enter the second string: world
Resulting string: hollo werld
```

DATE:10/10/2024

17. Sort dictionary in ascending and descending order.

PROGRAM

```
dis = { "banana": 10,
         "grape": 5,
        "pineapple": 20,
        "watermelon": 15 }
print(dis)
print("Ascending order:")
res = dict(sorted(dis.items()))
print(res)
print("Descending order:")
res = dict(sorted(dis.items(), reverse=True))
print(res)
```

OUTPUT

```
{'banana': 10, 'grape': 5, 'pineapple': 20, 'watermelon': 15}

Assending order:
{'banana': 10, 'grape': 5, 'pineapple': 20, 'watermelon': 15}

Decending order:
{'watermelon': 15, 'pineapple': 20, 'grape': 5, 'banana': 10}
```

DATE:18/10/2024

18. Merge two dictionaries.

PROGRAM

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

OUTPUT

```
{1: 'onion', 2: 'carrot', 3: 'tomato'}
{1: 'onion', 2: 'carrot', 3: 'tomato'}
```

DATE: 15/10/2024

19. Find the gcd of 2 numbers

PROGRAM

```
import math

x=int(input("Enter first number:"))
y=int(input("Enter second number:"))
print("GCD=",math.gcd(x,y))
```

OUTPUT

```
Enter first number:78
Enter second number:54
GCD= 6

Enter first number:789
Enter second number:765
GCD= 3
```

DATE: 22/10/2024

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

PROGRAM

```
li=[int(i) for i in input("Enter integers:").split()]
new_list=[i for i in li if i%2 != 0]
print("List after removing even numbers:",new_list)
```

OUTPUT

```
Enter integers:24 4 1 23 55
List after removing even numbers: [1, 23, 55]

Enter integers:3 47 27 2 23 88 478 9 79 90 44
List after removing even numbers: [3, 47, 27, 23, 9, 79]
```

COURSE OUTCOME 2

DATE: 08/10/2024

1.write a program to find factorial of a number

PROGRAM

```
n=int(input("Enter a number:"))
f=1
for i in range(n,1,-1):
        f=f*i
print("Factorial : ",f)
```

OUTPUT

```
Enter a number:6
Factorial :  720

Enter a number:7
Factorial : 5040
```

DATE: 08/10/2024

2.write a program to display the first n numbers of the Fibonacci series

PROGRAM

```
n=int(input("Enter number of elements:"))
first=0
sec=1
print(first)
print(sec)
for i in range(0,n-2):
        third=first+sec
        print(third)
        first=sec
        sec=third
```

OUTPUT

```
Enter number of elements:10
0
1
1
2
3
5
8
13
21
34

Enter number of elements:6
0
1
1
2
3
5
```

DATE:18/10/2024

3. Find the sum of all items in a list?

PROGRAM

```
l=[int(i) for i in input("Enter List : ").split()]
print("Sum : ",sum(l))
```

OUTPUT

```
Enter List : 7 6 2 9
Sum :  24

Enter List : 67 43 7 2
Sum :  119
```

DATE:18/10/2024

4. Generate a list of four digit numbers in a given range with all their digits even and the number is a perfect square?

PROGRAM

```
start = int(input("Enter the stating digits : "))
end = int(input("Enter the ending digits : "))
for i in range(int(start ** 0.5), int(end ** 0.5) + 1):
    square = i * i
    if start <= square <= end:
        if all(int(digit) % 2 == 0 for digit in str(square)):
            print(square)
```

OUTPUT

```
Enter the stating digits :1000
Enter the ending digits :9999
4624
6084
6400
8464

Enter the stating digits : 4000
Enter the ending digits : 6000
4624
```

DATE:08/10/2024

5. Display the given pyramid with step number accepted from user. Eg: N=4
1
2 4
3 6 9
4 8 12 16

PROGRAM

```
n=int(input("Enter Number:"))
for i in range(1, n+1):
        for j in range(1,i+1):
                print(i*j,end=' ')
        print()
```

OUTPUT

Enter Number:5
1
2 4
3 6 9
4 8 12 16
5 10 15 20 25

Enter Number:4
1
2 4
3 6 9
4 8 12 16

DATE:21/10/2024

6. Count the number of characters (character frequency) in a string?

PROGRAM

```
input_string = input("Enter a string: ")
char_frequency = {}
for char in input_string:
    if char in char_frequency:
        char_frequency[char] += 1
    else:
        char_frequency[char] = 1
for char, count in char_frequency.items():
    print(f"'{char}': {count}")
```

OUTPUT

```
Enter a string: hello world
'h': 1
'e': 1
'l': 3
'o': 2
' ': 1
'w': 1
'r': 1
'd': 1


Enter a string: programming
'p': 1
 'r': 2
 'o': 1
 'g': 2
 'a': 1
 'm': 2
 'i': 1
 'n': 1
```

DATE: 22/10/2024

7. Add ‘ing’ at the end of a given string. If it already ends with ‘ing’, then add ‘ly’?

PROGRAM

```
input_string = input("Enter a string: ")
if len(input_string) >= 3:
        if input_string.endswith("ing"):
                result = input_string + "ly"
        else:
                result = input_string + "ing"
else:
       result = input_string
print("Modified string:", result)
```

OUTPUT

```
Enter a string: Talk
Talking

Enter a string: Sleeping
Sleepingly
```

DATE: 29/10/2024

8. Accept a list of words and return length of longest word?

PROGRAM

```
words = input("Enter a list of words (separated by spaces): ").split()

longest_word = max(words, key=len)

print("Length of the longest word:", len(longest_word))
```

OUTPUT

```
Enter a list of words: apple banana mango orange
Length of the longest word: 6


Enter a list of words: cat dog elephant giraffe
Length of the longest word: 8
```

DATE: 29/10/2024

9. Construct following pattern using nested loop

```
*
* *
* * *
* * * *
* * * * *
* * * *
* * *
* *
*
```

PROGRAM

```
for i in range(0,4):
        for j in range(1,i+2):
                print("*",end=' ')
        print()
for i in range(5,0,-1):
        for j in range(1,i+1):
                print("*",end=' ')
        print()
```

OUTPUT

```
*
* *
* * *
* * * *
* * * * *
* * * *
* * *
* *
*
```

DATE: 29/10/2024

10. Generate all factors of a number?

PROGRAM

```
number = int(input("Enter a number: "))
factors = []

for i in range(1, number + 1):
    if number % i == 0:
        factors.append(i)

print("Factors of", number, "are:", factors)
```

OUTPUT

```
Enter a number: 20
Factors of 12 are: [1, 2, 4, 5, 10, 20]


Enter a number: 36
Factors of 36 are: [1, 2, 3, 4, 6, 9, 12, 18, 36]
```

DATE:29/10-2024

11. Write lambda functions to find area of square, rectangle and triangle?

PROGRAM

```
area_square = lambda side: side ** 2
area_rectangle = lambda length, width: length * width
area_triangle = lambda base, height: 0.5 * base * height
side = int(input("Enter the side : "))
length = int(input("Enter the length : "))
width = int(input("Enter the width : "))
base = int(input("Enter the base : "))
height = int(input("Enter the height : "))
print(f"Area of square: {area_square(side)}")
print(f"Area of rectangle: {area_rectangle(length, width)}")
print(f"Area of triangle: {area_triangle(base, height)}")
```

OUTPUT

```
Enter the side: 4
Enter the length: 5
Enter the width: 3
Enter the base: 6
Enter the height: 8
Area of square: 16
Area of rectangle: 15
Area of triangle: 24.0

Enter the side: 7
Enter the length: 10
Enter the width: 4
Enter the base: 12
Enter the height: 5
Area of square: 49
Area of rectangle: 40
Area of triangle: 30.0
```