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
- Customer 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: 3/10/2024

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

PROGRAM

```
year1=int(input("enter starting year "))
year2=int(input("enter final year "))
for x in range(year1,year2):
  if (x%4==0 and x%100!=0 )or x%400==0:
    print("leap year ",x)
```

OUTPUT

```
enter starting year 2020
enter final year 2030
leap year  2020
leap year  2024
leap year  2028

enter starting year 2020
enter final year 2030
leap year  2020
leap year  2024
leap year  2028

enter starting year 2008
enter final year 2018
leap year  2008
leap year  2012
leap year  2016
```

DATE: 3/10/2024

3. List comprehensions:
(a).Generate positive list of numbers from a given list of integers

PROGRAM

```
l=input("enter list of integers seperated by spaces ")
l1=[int(num) for num in l.split()]
pl=[num for num in l1 if num>0]
print("list of positive numbers are ",pl)
```

OUTPUT

```
enter list of integers seperated by spaces 2 4 5 -9 8 -10 4
list of positive numbers are  [2, 4, 5, 8, 4]

enter list of integers seperated by spaces 10 5 -8 2 -7 9
list of positive numbers are  [10, 5, 2, 9]
```

(b).Square of N numbers

PROGRAM

```
l=input("enter list of integers seperated by spaces ")
l1=[int(num) for num in l.split()]
print("square of numbers")
l2=[(num*num) for num in l1]
print(l2)
```

OUTPUT

```
enter list of integers seperated by spaces 2 4 6 8 10
square of numbers
[4, 16, 36, 64, 100]

enter list of integers seperated by spaces 1 3 5 6 9
square of numbers
[1, 9, 25, 36, 81]
```

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

PROGRAM

```
l=input("enter a word ")
l1=[x for x in l]
print(l1)
print("vowels")
l2=["a","e","i","o","u","A","E","I","O","U"]
l3=[x for x in l1 if x in l2]
print(l3)
```

OUTPUT

```
enter a word occurrences
['o', 'c', 'c', 'u', 'r', 'r', 'e', 'n', 'c', 'e', 's']
vowels
['o', 'u', 'e', 'e']

enter a word malayalam
['m', 'a', 'l', 'a', 'y', 'a', 'l', 'a', 'm']
vowels
['a', 'a', 'a', 'a']
```

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

PROGRAM

```
word=input("Enter a word : ")
ordinal_values = [ord(char) for char in word]
print(f"The ordinal values of the characters in the word '{word}' are: {ordinal_values}")
```

OUTPUT

```
Enter a word : apple
The ordinal values of the characters in the word 'apple' are: [97, 112, 112, 108, 101]

Enter a word : file
The ordinal values of the characters in the word 'file' are: [102, 105, 108, 101]
```

DATE: 8/10/2024

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

PROGRAM

```
l=input("Enter a line : ")
words = l.split()
res = {}
for word in words:
    word = word.lower()
    if word in res:
        res[word] += 1
    else:
        res[word] = 1
for word, count in res.items():
    print(f"'{word}': {count}")
```

OUTPUT

```
Enter a line : apple orange banana apple
'apple': 2
'orange': 1
'banana': 1

Enter a line : banana orange banana orange mango
'banana': 2
'orange': 2
'mango': 1
```

DATE: 8/10/2024

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

PROGRAM

```
u=input("enter list of integers seperated by spaces ")
numbers=u.split()
#numbers=[int(num) for num in u.split()]
result=[]
for num in numbers:
  numbers=int(num)
  if numbers > 100:
    result.append("over")
  else:
    result.append(numbers)
print("new list")
print(result)
```

OUTPUT

```
enter list of integers seperated by spaces 50 20 41 150 8 200 55
new list
[50, 20, 41, 'over', 8, 'over', 55]

enter list of integers seperated by spaces 24 30 150 77 160 55 200
new list
[24, 30, 'over', 77, 'over', 55, 'over']
```

DATE: 8/10/2024

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

PROGRAM

```
import math
l=[i for i in input("Enter List : ").split()]
count=0
for i in l:
    count+= i.lower().count('a')
print("Count of Letter A : ",count)
```

OUTPUT

```
Enter List : ananya
Count of Letter A :  3

Enter List : basil
Count of Letter A :  1
```

DATE: 8/10/2024

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

```
l1=[int(i) for i in input("Enter List 1 : ").split()]
l2=[int(i) for i in input("Enter List 2 : ").split()]
if len(l1) == len(l2):
   print("Length is same.")
else:
   print("Length is not same!")
if sum(l1) == sum(l2) :
   print("Sum of Lists are equal.")
else:
   print("Sum is not equal!")
c=set(l1).intersection(set(l2))
if len(c) != 0:
   print("Values : ",c)
else:
   print("No common elements!")
```

OUTPUT

```
Enter List 1 : 4 5 6 7 8 9
Enter List 2 : 4 5 6
Length is not same!
Sum is not equal!
Values :  {4, 5, 6}

Enter List 1 : 4 5 6 7 8
Enter List 2 : 8 7 6 5 4
Length is same.
Sum of Lists are equal.
Values :  {4, 5, 6, 7, 8}
```

DATE: 8/10/2024

8. Get a string from an input string where all occurrences of first character replaced with ‘$’,except first character[eg: onion -> oni$n]

PROGRAM

```
l=input("Enter a String : ")
f=l[0]
l1=l[1:].replace(f,'$')
print("New String : ",f+l1)
```

OUTPUT

```
Enter a String : malayalam
New String :  malayala$

Enter a String : tomato
New String :  toma$o
```

DATE: 10/10/2024

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

PROGRAM

```
s=input("Enter a String : ")
f=s[0]
l=s[-1:]
print("New String : ",l+s[1:-1]+f)
```

OUTPUT

```
Enter a String : welcome
New String :  eelcomw

Enter a String : update
New String :  epdatu
```

DATE: 10/10/2024

10. Accept the radius from user and find area of circle.

PROGRAM

```
r=int(input("enter radius "))
a=3.14*r*r
print("area of circle ",a)
```

OUTPUT

```
enter radius 10
area of circle  314.0

enter radius 15
area of circle  706.5
```

DATE: 10/10/2024

11. Program to find largest among 3 number

PROGRAM

```
a=int(input("enter num 1 "))
b=int(input("enter num 2 "))
c=int(input("enter num 3 "))
if a > b and a > c:
  print(a ,"is greater")
elif b > a and b > c:
  print(b ,"is greater")
elif c > a and c > b:
  print(c ,"is greater")
else:
  print("all are equal")
```

OUTPUT

```
enter num 1 24
enter num 2 12
enter num 3 10
24 is greater

enter num 1 30
enter num 2 48
enter num 3 20
48 is greater

enter num 1 20
enter num 2 11
enter num 3 50
50 is greater
```

DATE: 10/10/2024

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

PROGRAM

```
file=input("Enter File Name : ")
temp=file.split(".")
ext= temp[-1] if len(temp) > 1 else ""
print("Extension : ",ext)
```

OUTPUT

```
Enter File Name : file.txt
Extension :  txt

Enter File Name : img.jpg
Extension :  jpg
```

DATE: 10/10/2024

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

PROGRAM

```
l1=[i for i in input("enter the colors in list1: ").split()]
print("list")
print(l1)
print("first color: ",l1[1])
print("last color: ",l1[-1])
```

OUTPUT

```
enter the colors in list1: orange yellow green blue red
list
['orange', 'yellow', 'green', 'blue', 'red']
first color:  orange
last color:  red

enter the colors in list1: green blue red yellow black
list
['green', 'blue', 'red', 'yellow', 'black']
first color:  green
last color:  black
```

DATE: 15/10/2024

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

PROGRAM

```
x=int(input("Enter an Integer : "))
n1 = int(f"{x}") # nn
n2 = int(f"{x}{x}") # nn
n3 = int(f"{x}{x}{x}") #nnn
print(n1,"+",n2,"+",n3," = ",n1+n2+n3)
```

OUTPUT

```
Enter an Integer : 4
4 + 44 + 444  =  492

Enter an Integer : 2
2 + 22 + 222  =  246
```

DATE: 15/10/2024

15. Print out all colors from color-list1 not contained in color-list2.

PROGRAM

```
list1=[i for i in input("enter the colors in list1: ").split()]
list2=[i for i in input("enter the colors in list2: ").split()]
result=[i for i in list1 if i not in list2]
print("colors in list1 not in list2 ",result)
```

OUTPUT

```
enter the colors in list1: orange apple pineapple blueberry grapes
enter the colors in list2: apple orange banana
colors in list1 not in list2  ['pineapple', 'blueberry', 'grapes']

enter the colors in list1: black blue yellow orange
enter the colors in list2: yellow pink orange
colors in list1 not in list2  ['black', 'blue']
```

DATE: 15/10/2024

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

PROGRAM

```
s1=input("Enter String 1 :")
s2=input("Enter String 2 :")
new1=s1[0]+s2[1]+s1[2:]
new2=s2[0]+s1[1]+s2[2:]
print("S1 After Swap : ",new1,"\nS2 After Swap : ",new2)
```

OUTPUT

```
Enter String 1 :mango
Enter String 2 :orange
S1 After Swap :  mrngo
S2 After Swap :  oaange

Enter String 1 :drive
Enter String 2 :vehicle
S1 After Swap :  deive
S2 After Swap :  vrhicle
```

DATE: 15/10/2024

17. Sort dictionary in ascending and descending order.

PROGRAM

```
d={"apple":10,"orange":20,"banana":5,"kiwi":2}
print("dictionary ",d)
aresult=dict(sorted(d.items()))
dresult=dict(sorted(d.items(),reverse=True))
print("dictionary in ascending order ",aresult)
print("dictionary in descending order ",dresult)
```

OUTPUT

```
dictionary  {'apple': 10, 'orange': 20, 'banana': 5, 'kiwi': 2}
dictionary in ascending order  {'apple': 10, 'banana': 5, 'kiwi': 2, 'orange': 20}
dictionary in descending order  {'orange': 20, 'kiwi': 2, 'banana': 5, 'apple': 10}
```

DATE: 15/10/2024

18. Merge two dictionaries.

PROGRAM

```
d1={"apple":10,"orange":20,"banana":5,"kiwi":2}
d2={"pineapple":50,"mango":30}
print(d1)
print(d2)
d1.update(d2)
print(d1)
print(d1|d2)
```

OUTPUT

```
{'apple': 10, 'orange': 20, 'banana': 5, 'kiwi': 2}
{'pineapple': 50, 'mango': 30}
{'apple': 10, 'orange': 20, 'banana': 5, 'kiwi': 2, 'pineapple': 50, 'mango': 30}
{'apple': 10, 'orange': 20, 'banana': 5, 'kiwi': 2, 'pineapple': 50, 'mango': 30}
```

DATE: 22/10/2024

19. Find gcd of 2 numbers.

PROGRAM

```
import math
x=int(input("enter num1 "))
y=int(input("enter num2 "))
print("gcd is ",math.gcd(x,y))
```

OUTPUT

```
enter num1 10
enter num2 25
gcd is  5

enter num1 18
enter num2 9
gcd is  9
```

DATE: 22/10/2024

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

PROGRAM

```
ol=[int(i) for i in input("enter the integers ").split()]
print("list before ",ol)
newlist=[i for i in ol if i%2!=0]
print("list after removing even numbers ",newlist)
```

OUTPUT

```
enter the integers 3 5 8 9 10 4 88
list before  [3, 5, 8, 9, 10, 4, 88]
list after removing even numbers  [3, 5, 9]

enter the integers 9 7 8 11 4 77 16 5
list before  [9, 7, 8, 11, 4, 77, 16, 5]
list after removing even numbers  [9, 7, 11, 77, 5]
```

COURSE OUTCOME 2

DATE: 22/10/2024

1. Program to find the factorial of a number

PROGRAM

```
n=int(input("enter number "))
fact=1
i=1
while i <= n:
  fact=fact*i
  i=i+1
print("factorial of ",n,"is ",fact)
```

OUTPUT

```
enter number 5
factorial of  5 is  120

enter number 4
factorial of  4 is  24
```

DATE: 22/10/2024

2. Generate Fibonacci series of N terms

PROGRAM

```
n = int(input("Enter the number of terms: "))
a, b = 0, 1
fibonacci_series = []
for i in range(n):
    fibonacci_series.append(a)
    a, b = b, a + b
print(f"Fibonacci series of {n} terms: {fibonacci_series}")
```

OUTPUT

```
Enter the number of terms: 5
Fibonacci series of 5 terms: [0, 1, 1, 2, 3]

Enter the number of terms: 10
Fibonacci series of 10 terms: [0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
```

DATE: 24/10/2024

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 : 5 4 7 9 10 6
Sum :  41

Enter List : 5 10 15 20 25
Sum :  75
```

DATE: 24/10/2024

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

```
import math

def is_all_digits_even(num):
    while num > 0:
        digit = num % 10
        if digit % 2 != 0:
            return False
        num //= 10
    return True

def even_digit_perfect_squares(start, end):
    even_digit_squares = []
    for num in range(start, end + 1):
        if num >= 1000 and num <= 9999 and is_all_digits_even(num):
            root = int(math.sqrt(num))
            if root * root == num:
                even_digit_squares.append(num)
    return even_digit_squares

start = 1000
end = 9999

even_digit_squares = even_digit_perfect_squares(start, end)
print(even_digit_squares)
```

OUTPUT

```
[4624, 6084, 6400, 8464]
```

DATE: 24/10/2024

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

```
for i in range(1,5):
  for j in range(1,i+1):
   print(i*j,end=" ")
  print(" ")
```

OUTPUT

```
1
2 4
3 6 9
4 8 12 16
```

DATE: 24/10/2024

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

PROGRAM

```
from collections import Counter
s = input("Enter a string: ")
char_frequency = Counter(s)
print("Character Frequency:")
for char, freq in char_frequency.items():
    print(f"'{char}': {freq}")
```

OUTPUT

```
Enter a string: welcome
Character Frequency:
'w': 1
'e': 2
'l': 1
'c': 1
'o': 1
'm': 1

Enter a string: hello
Character Frequency:
'h': 1
'e': 1
'l': 2
'o': 1
```

DATE: 24/10/2024

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

PROGRAM

```
s=input("Enter a string :")
a=s[-3:]
if a=='ing':
   print(s+'ly')
else:
   print(s+'ing')
```

OUTPUT

```
Enter a string :cry
crying

Enter a string :making
makingly
```

DATE: 29/10/2024

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

PROGRAM

```
s=[i for i in input("Enter some words :").split()]
print(len(max(s, key=len)))
```

OUTPUT

```
Enter some words :programming language
11

Enter some words :user input
5
```

DATE: 29/10/2024

9. Construct following pattern using nested loop

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

PROGRAM

```
for i in range(5):
    for j in range(i + 1):
        print("*", end=" ")
    print()

for i in range(5):
    for j in range(5-i-1):
        print("*", end=" ")
    print()
```

OUTPUT

```
Enter the size: 5
*
* *
* * *
* * * *
* * * * *
* * * *
* * *
* *
*
```

DATE: 29/10/2024

10. Generate all factors of a number.

PROGRAM

```
def factors(a):
    for i in range(1,a+1):
        if a%i == 0:
            print(i)
a=int(input("Enter a number :"))
factors(a)
```

OUTPUT

```
Enter a number :8
1
2
4
8

Enter a number :12
1
2
3
4
6
12
```

DATE: 29/10/2024

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

PROGRAM

```
area1=lambda a :a*a
area2=lambda l,b :l*b
area3=lambda b,h :0.5*b*h
s=int(input("enter side of square "))
print("area of square= ",area1(s))
l=int(input("enter length of rectangle "))
b=int(input("enter breadth of rectangle "))
print("area of rectangle= ",area2(l,b))
p=int(input("enter base of triangle "))
h=int(input("enter height of triangle "))
print("area of triangle= ",area3(p,h))
```

OUTPUT

```
enter side of square 4
area of square=  16
enter length of rectangle 10
enter breadth of rectangle 2
area of rectangle=  20
enter base of triangle 8
enter height of triangle 10
area of triangle=  40.0

enter side of square 8
area of square=  64
enter length of rectangle 6
enter breadth of rectangle 12
area of rectangle=  72
enter base of triangle 5
enter height of triangle 12
area of triangle=  30.0
```