COURSE OUTCOME 1

DATE : 18-09-2023

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:

a. 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

b. **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

c. 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.

d. 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

e. 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 with 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.

f. 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

g. 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 analyze 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.

COURSE OUTCOME 1

DATE : 18-09-2023

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

PROGRAM

```
y1=int(input("enter starting year:"))
y2=int(input("enter last year:"))
for year in range(y1,y2):
 if(year%4==0)and(year%100!=0):
 print(year)
```

OUTPUT

```
enter starting year:2002
enter last year:2025
2004
2008
2012
2016
2020
2024
```

COURSE OUTCOME 1

DATE : 18-09-2023

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

PROGRAM

```
l1=[1,-1,-2,2,-3,3,4,5]
l2=[i for i in l1 if i>0]
print("Positive integers:",l2)
```

OUTPUT

Positive integers: [1, 2, 3, 4, 5]

    b. Square of N numbers.

PROGRAM

```
n=int(input("Enter limit:"))
l=[i*i for i in range (n)]
l
```

OUTPUT

Enter limit:5
[0, 1, 4, 9, 16]

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

PROGRAM

```
a=input("Enter a word:")
v=[i for i in a if i in 'aeiouAEIOU']
v
```

OUTPUT

```
Enter a word:education
['e', 'u', 'a', 'i', 'o']
```

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

PROGRAM

```
n=int(input("Enter limit:"))
l=[i*i for i in range (n)]
l
```

OUTPUT

```
Enter limit:5
[0, 1, 4, 9, 16]
```

COURSE OUTCOME 1

DATE : 18-09-2023

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

PROGRAM

```
text =      input    ("Enter a line of text: ")
words=text.lower().split()
word_count={word:words.count(word) for word in words}
for word,count in word_count.items():
    print(f" '{word}' : {count}")
```

OUTPUT

```
Enter a line of text: she is playing
'she': 1
'is': 1
'playing': 1
```

COURSE OUTCOME 1

DATE : 18-09-2023

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

PROGRAM

```
list=[]
n=int(input("enter the number:"))
for i in range (0, n):
  nu=int(input())
  if nu>100:
    list.append('over')
  else:
      list.append(nu)
print(list)
```

OUTPUT

```
enter the number:4
9
7
111
160


[9, 7, 'over', 'over']
```

COURSE OUTCOME 1

DATE : 20-09-2023

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

PROGRAM

```
name=['anu' 'aparna ' 'priya'    ]
for i in name:
        print("a","occurs in",i,i.count('a'),"times")
```

OUTPUT

a occurs in anu 1 times

a occurs in aparna 3 times

a occurs in priya 1 times

COURSE OUTCOME 1

DATE : 20-09-2023

7. Enter 2 lists of integers. Check
    a. Whether lists are of the same length.
    b. Whether the list sums to the same value.
    c. Whether any value occurs in both.

PROGRAM

```
l1=[]
l2=[]
n1=int(input("Enter the size of list 1 : "))
print("Enter list 1 : ")
for i in range(n1):
  i=int(input())
  l1.append(i)

n2=int(input("Enter the size of list 2 : "))
print("Enter list 2 : ")
for i in range(n2):
  i=int(input())
  l2.append(i)

if n1==n2:
  print("Lists are of same length")
else:
  print("Lists are of different length")


if sum(l1)==sum(l2):
  print("Sum of lists are of same")
else:
  print("Sum of lists are different")

res=[i for i in l1 if i in l2]
print("Common elements : ",res)
```

OUTPUT

```
Enter the size of list 1 : 3
Enter list 1 :
```

2

5

6

Enter the size of list 2 : 3

Enter list 2 :

9

5

7

Lists are of same length

Sum of lists are different

Common elements : [5]

COURSE OUTCOME 1

DATE : 20-09-2023

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

   PROGRAM

```
str1=input("Enter a word : ")
f_c=str1[0]
str1=str1.replace(f_c,'$')
str1=f_c+str1[1:]
print(str1)
```

   OUTPUT

```
Enter a word : onion
oni$n
```

COURSE OUTCOME 1

DATE : 20-09-2023

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

PROGRAM

```
name=input("Enter a name : ")
name[-1]+name[1:-1]+name[0]
```

OUTPUT

Enter a name : geethu

'ueethg’

COURSE OUTCOME 1

DATE : 20-09-2023

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

PROGRAM

```
r=float(input("Enter the radius : "))
print("Area : ",3.14*r*r)
```

OUTPUT

```
Enter the radius : 62
Area : 12068.50
```

COURSE OUTCOME 1

DATE : 27-09-2023

11. Find the biggest of 3 numbers entered.

PROGRAM

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

OUTPUT

Enter the first number : 3
Enter the second number : 2
Enter the third number : 8
8 is the largest number

Enter the first number : 2
Enter the second number : 5
Enter the third number : 3
5 is the largest number

Enter the first number : 6
Enter the second number : 4
Enter the third number : 3
6 is the largest number

COURSE OUTCOME 1

DATE : 27-09-2023

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

PROGRAM

```
f_name=input("Enter a filename : ")
f_ext=f_name.split(".")
print("Extension : ",f_ext[-1])
```

OUTPUT

```
Enter a filename : index.html
Extension : html
```

COURSE OUTCOME 1

DATE : 27-09-2023

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

PROGRAM

```
color_input = input("Enter a list of colors separated by commas: ")

colors = color_input.split(',')


colors = [color.strip() for color in colors]


if len(colors) >= 2:

    print(f"First color: {colors[0]}")
    print(f"Last color: {colors[-1]}")
else:
    print("Please enter atleast two colors separated by commas." )
```

OUTPUT

```
Enter a list of colors separated by commas: red,blue,black,green
First color: red
Last color: green
```

COURSE OUTCOME 1

DATE : 27-09-2023

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

PROGRAM

```
n=int(input("Enter a number : "))
print("Result :",n+n*11+n*111)
```

OUTPUT

```
Enter a number : 7
Result:861
```

COURSE OUTCOME 1

DATE : 27-10-2023

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

PROGRAM

```
c1=["Red","Black","Green","Blue","Violet","Purple"]
c2=["Pink" ,"Black","Cyan""Blue","Magenta","Brown"]
s=[i    for i in c2 if i not in c1]
print("Colors from color-list 1 not in colo-list 2 : ",s)
```

OUTPUT

Colors from color-list 1 not in colo-list 2 : ['Pink', 'Cyan', 'Magenta', 'Brown']

COURSE OUTCOME 1

DATE : 04-10-2023

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 : ")
s3=s2[0]+s1[1:]+" "+s1[0]+s2[1:]
print("Swapped string :",s3)
```

OUTPUT

```
Enter string 1 : python
Enter string 2 : lab
Swapped string : lython pab
```

COURSE OUTCOME 1

DATE : 04-10-2023

17. 17. Sort the dictionary in ascending and descending order.

PROGRAM

```
d = {'gokul'   : 10 ,'john'  : 20 'lechu'  : 30}
print("Ascending order : ",dict(sorted(d.items())))
print("Descending order : ",dict(sorted(d.items(), reverse=True)))
```

OUTPUT

```
Ascending order : {'gokul': 10, 'john': 20, 'lechu': 30}
Descending order : {'lechu': 30, 'john': 20, 'gokul': 10}
```

COURSE OUTCOME 1

DATE : 04-10-2023

18. Merge two dictionaries.

PROGRAM

```
d1 = {'aparna'    :15 'anu'   : 20 ,'diya' : 30}
d2 = {'archana':25 'anjana': 40 ,'lekshmi': 50}
print("Merged dictionaries :",d1|d2)
```

OUTPUT

```
Merged dictionaries : {'aparna': 15, 'anu': 20, 'diya': 30, 'archana': 25, 'anjana': 40,
'lekshmi': 50}
```

COURSE OUTCOME 1

DATE : 04-10-2023

19. Find gcd of 2 numbers.

PROGRAM

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

OUTPUT

```
Enter the first number : 8
Enter the second number : 24
GCD : 8
```

COURSE OUTCOME 1

DATE : 04-10-2023

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

PROGRAM

```
l=[]
n=int(input("Enter the size of the list : "))
print("Enter elements : ")
for i in range(n):
  i=int(input())
  l.append(i)

for i in l:
  if i%2==0:
    l.remove(i)

print("List after removal of even numbers : ",l)
```

OUTPUT

```
Enter the size of the list : 5
Enter elements :
1
2
3
4
5
List after removal of even numbers : [1, 3, 5]
```

COURSE OUTCOME 2

DATE : 09-10-2023

1. Program to find the factorial of a number.

PROGRAM

```
def fact(x):
  if x==1:
    return 1
  else:
    return x*fact(x-1)
x=int(input("Enter the number : "))
print("Factorial : ",fact(x))
```

OUTPUT

```
Enter the number : 5
Factorial : 120
```

COURSE OUTCOME 2

DATE : 09-10-2023

2. Generate Fibonacci series of N terms.

PROGRAM

```
n=int(input("Enter the no of terms : "))
a=0
b=1
c=a+b
print(a)
print(b)
print(c)
for i in range(3,n):
  a=b
  b=c
  c=a+b
  print(c)
```

OUTPUT

```
Enter the no of terms : 7
0
1
1
2
3
5
8
```

COURSE OUTCOME 2

DATE : 09-10-2023

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

PROGRAM

```
l=[]
n=int(input("Enter the size of the list : "))
print("Enter elements : ")
for i in range(n):
  i=int(input())
  l.append(i)

print("Sum : ",sum(l))
```

OUTPUT

```
Enter the size of the list : 5
Enter elements :
1
2
3
4
5
Sum:15
```

COURSE OUTCOME 2

DATE : 09-10-2023

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

```
s=[]
for i in range(1000,10000):
    if all(int(x)%2==0 for x in str(i)):
        if int(i**0.5)**2==i:
            s.append(i)
print("List of numbers : ",s)
```

OUTPUT

List of numbers : [4624, 6084, 6400, 8464]

COURSE OUTCOME 2

DATE : 09-10-2023

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

PROGRAM

```
N = int(input("Enter the number of steps for the pyramid: "))

for i in range(1, N + 1):
    for j in range(1, i + 1):

        value = i * j

        print(value, end=" ")

    print()
```

OUTPUT

```
Enter the number of steps for the pyramid: 4
1
2 4
3 6 9
4 8 12 16
```

COURSE OUTCOME 2

DATE : 11-10-2023

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

PROGRAM

```
input_string = input("Enter a string: ")
char_count = {}
for char in input_string:
    char_count[char] = char_count.get(char, 0) + 1
for char, count in char_count.items():
 print(f"'{char}': {count}")
```

OUTPUT

```
Enter a string: python
'p': 1
'y': 1
't': 1
'h': 1
'o': 1
'n': 1
```

COURSE OUTCOME 2

DATE : 11-10-2023

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

PROGRAM

```
str1=input("Enter a string : ")
if str1.endswith('ing'):
  str2=str1+'ly'
else:
  str2=str1+'ing'
print(str2)
```

OUTPUT

```
Enter a string : play
playing

Enter a string : playing
playingly
```

COURSE OUTCOME 2

DATE : 11-10-2023

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

PROGRAM

```
n = int(input("Enter the size of the list: "))
a = [input("Enter word: ") for _ in range(n)]


temp = max(a, key=len)

print("Word with max length is", temp, "Its length is", len(temp))
```

OUTPUT

```
Enter the size of the list: 3
Enter word: python
Enter word: programming
Enter word: lab
Word with max length is programming Its length is 11
```

COURSE OUTCOME 2

DATE : 11-10-2023

9. Construct following pattern using nested loop

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

PROGRAM

```
n=int(input("Enter the no of rows : "))
for i in range(1,n+1):
  print('*'*i)
for i in range(n-1,0,-1):
  print('*'*i)
```

OUTPUT

```
Enter the no of rows : 5
*
**
***
****
*****
****
***
**
*
```

COURSE OUTCOME 2

DATE : 11-10-2023

10. Generate all factors of a number.

PROGRAM

```
def facts(x):
    return [i for i in range(1, x + 1) if x % i == 0]

n = int(input("Enter a number: "))
factors = facts(n)

print(f"Factors of {n} are: {factors}")
```

OUTPUT

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

COURSE OUTCOME 2

DATE : 11-10-2023

11. Write lambda functions to find the 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 the side of square : "))
print("Area of square : ",area1(s))

l=int(input("Enter the length of rectangle : "))
b=int(input("Enter the breadth of rectangle : "))
print("Area of rectangle : ",area2(l,b))

b=int(input("Enter the base of triangle : "))
h=int(input("Enter the height of triangle : "))
print("Area of triangle : ",area3(b,h))
```

OUTPUT

```
Enter the side of square : 15
Area of square : 225
Enter the length of rectangle : 14
Enter the breadth of rectangle : 12
Area of rectangle : 168
Enter the base of triangle : 35
Enter the height of triangle : 15
Area of triangle :  262.5
```