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, buildingexecutable, and debugging. IDEs are usually more feature-rich and include tools fordebugging, 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 IDEand 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 aregenerally 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 IDLEtool 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 emphasison 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 andmachine 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.

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

Program

```
x=int(input("enter the start year: "))
y=int(input("enter the end year: "))
for year in range(x,y):
if(0==year%4) and (0!=year%100)or(0==year%400):
print(year)
```

Output

enter the start year: 1999

enter the end year: 2023

2000 2004 2008 2012 2016 2020

3. List comprehensions:

a) **Generate positive list of numbers from a given list of integers**
b) **Square of N numbers**
c) **From 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)
li=[1,2,3,4,5,6,-1,-2,-3]
l2=[i for i in li if i>=0]
print(l2)
```

Output

[1, 2, 3, 4, 5, 6]

```
b)
n=int(input("Enter the limit"))
li=[i*i for i in range(n)]
print(li)
```

Output

Enter the limit6

[0, 1, 4, 9, 16, 25]

```
c)
w=input("Enter a word:")
l1=[i for i in w if i in "aeiouAEIOU"]
print(l1)
```

Output

Enter a word:apple ['a', 'e']

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

Program

```
s=input("Enter a line: ")
l1=[]
l1=s.split()
c=[l1.count(i) for i in l1]
print(dict(zip(l1,c)))
```

Output

Enter a line: hello world world is pretty {'hello': 1, 'world': 2, 'is': 1, 'pretty': 1}

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

```
a=int(input("Enter the number of elements"))
list=[]
for i in range(a):
b=int(input("Enter the elements"))
if b<100:
list.append(b)
else:
list.append("over")
print(list)
```

Output

Enter the number of elements 5

Enter the elements 100

Enter the elements 110

Enter theelements 10

Enter the elements 20

Enter the elements 30

['over', 'over', 10, 20, 30]

Date:20/09/2023

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

Program

```
li=["karun","sarah","nathan"]
for i in li:
print("a occurs in",i,i.count("a"),"times")
```

Output

a occurs in hari 1 times a

occurs in sarah 2 times a

occurs in nathan 2 times

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

```
s1=0
s2=0
m=int(input("Enter the number of integers in the list: "))
list_a=[]
print("Enter the list: ")
for i in range(m):
c1=int(input())
s1=s1+c1
list_a.append(c1)
print(list_a)
n=int(input("Enter the number of in the second list: "))
list_b=[]
print("Enter the list: ")
for i in range(n):
c2=int(input()) s2=s2+c2
list_b.append(c2)
print(list_b)
if len(list_a)==len(list_b):
print("list_a and list_b have same length")
else:
print("lengths are not same")if
s1==s2:
print("sum of two lists are same")
else:
print("Sum is not equal")
```

```
check=any(item in list_a for item in list_b)if

check is True:

print("Yes")

c=[i for i in list_a if i in list_b]

print("common elements are: ",c)
```

Output

```
Enter the number of integers in the list: 5

 Enter the list: 2 [2] 3 [2, 3] 44 [2, 3, 44] 5 [2, 3,44, 5] 66 [2, 3, 44, 5, 66]

Enter the number of in the second list: 3

 Enter the list: 2 5 44
[2, 5, 44]
 lengths are not same
 Sum is not equal Yes
common elements are: [2, 44, 5]
```

8. Get a string from an input string where all occurrences of first character replacedwith '$', except first character.

[eg: onion -> oni$n]

Program

```
a=input("Enter the string:")
char = a[0]
a = a.replace(char, '$')
a = char + a[1:]
print(a)
```

Output

Enter the string:onion oni$n

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

Program

```
a=input("Enter the string:")
first_char = a[0]
last_char = a[-1]
b= last_char + a[1:-1] + first_char
print(b)
```

Output

Enter the string python

nythop

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

Program

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

Output

Enter the radius:2.5

19.625

Date:27/09/2023

11. Find 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")
else:
print(c,"is the largest")
```

Output

enter the first number 23

enter the second number 55

enter the third number 0

55 is thelargest

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

Program

```
a=input("Enter the filename")
b=a.split(".")
print("Extension of the file is:" + repr(b[-1]))
```

Output

Enter the filenamemy file.py

Extension of the file is:'py'

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

Program

```
l1=["red","white","black","green"]
print("First color of the list is",l1[0])
print("Last element of the list is",l1[-1])
```

Output

First color of the list is red

Last element of the list is green

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

Program

```
n=int(input("Enter the number:"))
nn=n*11
nnn=n*111
result=n+nn+nnn
print(f"{n}+{nn}+{nnn}={result}")
```

Output

Enter the number:3

3+33+333=369

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

Program

```
color_list1 = set(["red","orange","blue","pink"])
color_list2 = set(["purple","red","black","green","blue"])
print(color_list1.difference(color_list2))
```

Output

{'orange', 'pink'}

Date:04/10/2023

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

Program

```
def chars_mix_up(a, b):
new_a = b[:2] + a[2:]
new_b = a[:2] + b[2:]

return new_a + ' ' + new_b
print(chars_mix_up('dpk', 'van'))
```

Output

vak dpn

17. Sort dictionary in ascending and descending order.

Program

```
d = {'Manuel': 10, 'hari': 20,'basil': 30}
print("Ascending order : ",dict(sorted(d.items())))
print("Descending order : ",dict(sorted(d.items(),
```

Output

Ascending order : {'Manuel': 10, 'basil': 30, 'hari': 20}
Descending order : {'hari': 20, 'basil': 30, 'Manuel': 10}

18. Merge two dictionaries.

Program

```
dict1 = {'a': 10, 'b': 8}
dict2 = {'d': 6, 'c': 4}
print(dict1|dict2)
```

Output

{'a': 10, 'b': 8, 'd': 6, 'c': 4}

19. Find gcd of 2 numbers.

Program

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

Output

Enter the first number : 23

Enter the second number : 22

GCD : 1

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

Program

```
list=[10,1,8,21,89,32,23,59]
print("The old list: ")
print(list)
for i in list:
if i % 2==0:
list.remove(i)
print("List after removing even numbers:")
print(list)
```

Output

```
The old list: [10, 1, 8, 21, 89, 32, 23, 59]
 List after removing even numbers: [1, 8, 21, 89,32, 23, 59]
List after removing even numbers: [1, 21, 89, 32, 23, 59]
 List after removing even numbers: [1, 21, 89, 23, 59]
```

COURSE OUTCOME 2

Date:09/10/2023

1.Program to find the factorial of a number

Program

```
a=int(input("enter the number"))
fac=1
for i in range (1,a+1):
fac=fac*i
print(fac)
```

Output

enter the number 5
120

2.Generate Fibonacci series of N terms

Program

```
num = 10
n1, n2 = 0, 1
print("Fibonacci Series:", n1, n2, end=" ")
for i in range(2, num):
n3 = n1 + n2
n1 = n2
n2 = n3
print(n3, end=" ")

print()
```

Output

Fibonacci Series: 0 1 1 2 3 5 8 13 21 34

3.Find the sum of all items in a list

Program

```
lst = []
num = int(input('How many numbers: '))
for n in range(num):
numbers = int(input('Enter number '))
lst.append(numbers)
print("Sum of elements in given list is :", sum(lst))
```

Output

How many numbers: 5

Enter number 55

Enter number 44

Enter number 3

Enter number 6

Enter number 8

Sum of elements in given list is : 116

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

```
i=1
t=int(input("please enter the range till runs "))
for j in range(1,t+1):
while(i<=((j//2)+1)):
if(i**2==j):
print(j)
i+=1
i=1
```

Output

please enter the range till runs 3 1

5.Display the given pyramid with step number accepted from user. Eg: N=41

2 4

3 6 9

4 8 12 16

Program

```
rows = int(input("Enter number of rows: "))
number=1
for i in range(1,rows+1):
for j in range(1,i+1):
sqaure=i*j
print(i*j, end=" ")
print()
```

Output

Enter number of rows: 4
1
2 4
3 6 9
4 8 12 16

Date:11/10/2023

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

Program

```
def char_frequency(str1):
dict = {}
for n in str1:
keys = dict.keys()
if n in keys:
dict[n] += 1
else:
dict[n] = 1
return dict
print(char_frequency('hari'))
```

Output

{'h': 1, 'a': 1, 'r': 1, 'i': 1,}

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

Program

```
string = input() if
len(string) < 3:
print(string)
elif string[-3:] == 'ing':
print(string +  'ly') else:
print(string + 'ing')
```

Output

drowning drowningly

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

Program

```
a=[]
n= int(input("Enter the number of elements in list:"))for
x in range(0,n):
element=input("Enter element" + str(x+1) + ":")
a.append(element)
max1=len(a[0])
temp=a[0]
for i in a:
if(len(i)>max1):
max1=len(i)
temp=i
print("The word with the longest length is:")
print(temp)
```

Output

Enter the number of elements in list:3

Enter element1:Hari

Enter element2:niranjan

Enter element3:M

The word with the longest length is: niranjan

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

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

10. Generate all factors of a number.

Program

```
num = int(input("Enter a number : "))
count = 0
print("The factors of",num,"are ",end="")
for i in range(1, num+1, 1):
if(num % i == 0):
count = count + 1
print(i, end = "")
print("\nTotal factors of",num,":",count)
```

Output

Enter a number : 20

The factors of 20 are 1 2 4 5 10 20

Total factors of 20 : 6

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

Program

```
area_s=lambda a: a*a
area_r=lambda l,b: l*b
area_t=lambda b,h: 0.5*(b*h)
s=int(input("Enter the side of square : "))
print("Area of square : ",area_s(s))
l=int(input("Enter the length of rectangle : "))
b=int(input("Enter the breadth of rectangle : "))
print("Area of rectangle : ",area_r(l,b))
b=int(input("Enter the base of triangle : "))
h=int(input("Enter the height of triangle : "))
print("Area of triangle : ",area_t(b,h))
```

Output

Enter the side of square : 4

Area of square : 16

Enter the length of rectangle : 3

Enter the breadth of rectangle : 6

Area of rectangle : 18

Enter the base of triangle : 7

Enter the height of triangle : 9

Area of triangle : 31.5
