**Department of Computer Science and Engineering**

**Laboratory Manual**

**PYTHON PROGRAMMING**

**14CS704**

**NMAM INSTITUTE OF TECHNOLOGY**

*(An Autonomous Institution affiliated to VTU, Belgaum)*

(NBA Accredited, ISO 9001:2008 Certified)

Nitte – 574110, Karkala, Udupi District, Karnataka, India

**PYTHON PROGRAMMING**

**14CS704**

**VII**

**Software used**

1. Python3

2. Eclipse Neon

**Marks distribution**

**Internal Evaluation**

|  |  |
| --- | --- |
| **Evaluation Criteria** | **Marks** |
| CIE | 10 |
| Project | 20 |
| Test | 20 |

**SEE**

|  |  |
| --- | --- |
| **Evaluation Criteria** | **Marks** |
| Program write up | (5+5) |
| Program execution | (15+15) |
| Viva-Voce | 10 |

**Guidelines for mini project (if applicable)**

1. Mini project must be carried out by a team of two members.

2. Mini project must be carried out to solve any relevant general purpose programming problem.

3. Implement the mini project using the python concepts in a meaningful way.

4. Evaluation is based on problem addressed, use of the programming concepts and the user experience of the mini project.

**Prepared by:**

|  |  |  |
| --- | --- | --- |
| **Name of the faculty** | **Designation** | **Unit** |
| Mr.Venugopala P S | Associate Professor | UNIT-I |
| Mr.Ranjan Kumar H S | Assisatant Professor | UNIT-II |
| Mr.Pawan Hegde | Assistant Professor | UNIT-III |
| Mr.Vijaya Murari T. | Assistant Professor | UNIT-IV,UNIT-V |

**List of Experiments**

|  |  |  |
| --- | --- | --- |
| **Sl.No.** | **Title of the experiment** | **Page Number** |
|  | **UNIT-I** |  |
| 1. | Write a program which will find all such numbers which are divisible by 7 but are not a multiple of 5, between 2000 and 3200 (both included). | 5 |
| 2. | With a given integral number n, write a program to generate a dictionary that contains (i, i\*i) such that is an integral number between 1 and n (both included) and then the program should print the dictionary. | 5 |
| 3. | Write a program which accepts a sequence of comma-separated numbers from console and generate a list and a tuple which contains every number. | 6 |
| 4. | Write a program that accepts a comma separated sequence of words as input and prints the words in a comma-separated sequence after sorting them alphabetically. | 6 |
| 5. | Write a program that accepts sequence of lines as input and prints the lines after making all characters in the sentence capitalized | 7 |
| 6. | Write a program that accepts a sentence and calculate the number of letters and digits. | 7 |
| 7. | A website requires the users to input username and password to register. Write a program to check the validity of password input by users. | 8 |
| 8. | Write a Program to demonstrate the reading and writing of text to a file. | 9 |
|  | **UNIT-II** |  |
| 1. | Write a program to find second largest and second smallest number in a given list. | 9 |
| 2. | Write a program to count the number of occurrence of characters (character frequency) in a string and number of occurrence of words in a sentence. | 10 |
| 3. | Write a program to create a Caesar encryption. | 10 |
| 4. | Write a program to create a list by entering countries and respective capital and population. The program should accept the name of a country as an input and print the corresponding capital name and population as output. Otherwise, the program should print an appropriate message if the country is not found in the list. | 11 |
|  | **UNIT-III** |  |
| 1. | Write a binary search function which searches an item in a sorted list. The function should return the index of element to be searched in the list. | 12 |
| 2. | Write a program to count the number of occurrence of characters (character frequency) in a string and number of occurrence of words in a sentence. | 12 |
| 3. | With a given list [12, 24, 35, 24, 88, 120, 155, 88, 120, 155], write a program to print this list after removing all duplicate values with original order reserved. | 12 |
| 4. | Write a program using list comprehension to print the Fibonacci Sequence in comma separated form with a given n input by console. | 13 |
| 5. | Craps is a popular dice game played in casinos. Write a program to play a variation of the game | 13 |
| 6. | Design a class named Quadratic Equation for a quadratic equation ax 2 + bx + x = 0. The class contains:  ■ The private data fields a , b , and c that represent three coefficients.  ■ A constructor for the arguments for a , b , and c .  ■ Three get methods for a , b , and c .  ■ A method named getDiscriminant() that returns the discriminant, which is b 2 - 4ac.  ■ The methods named getRoot1() and getRoot2() for returning the two roots of the equation. | 14 |
| 7. | Design a class named Location for locating a maximal value and its location in a two-dimensional list. The class contains the public data fields like row, column, and maxValue that store the maximal value and its indexes in a two-dimensional list, with row and column as int types and maxValue as a float type.  Write the following method that returns the location of the largest element in a two-dimensional list. | 15 |
| 8. | Demonstrate the working of Operator Overloading in python | 16 |
|  | **UNIT-IV** |  |
| 1. | Write a program to demonstrate the creation of window, label and a button. | 17 |
| 2. | Write a program to demonstrate the processing of the button event. | 17 |
| 3. | Write a program that receives two numbers from the text fields and calculates the sum and then displays it in a message box. Generate an exception if the first number is less than the second. | 18 |
| 4. | Create a GUI that is having a text box to enter the name, radio button to select the gender, check-box to choose hobbies. Write a function that’s gets called on the button click to display these values as labels. | 19 |
| 5. | Write a program to demonstrate the concept of multithreading. | 21 |
| 6. | Write a program to demonstrate the concept of synchronizing the threads. | 22 |
| 7. | Write a program to demonstrate the concept of client server program. | 24 |
|  | **UNIT-V** |  |
| 1. | Write a program to create a database table EMPLOYEE. | 26 |
| 2. | Write a program which executes SQL *INSERT* statement to create a record in the EMPLOYEE table. | 27 |
| 3. | Write a program that updates all the records having SEX as **'M'**. Here, the AGE of all the males is increased by one year. | 28 |
| 4. | DELETE operation is required when you want to delete some records from your database. Following is the program to delete all the records from EMPLOYEE where AGE is more than 20. | 28 |
| 5. | Write sample program to demonstrate the concept of CGI. | 29 |
| 6. | Write a program to demonstrate the passing of information using GET method. | 29 |
| 7. | Simple FORM Example: GET Method | 30 |
| 8. | Write a program to demonstrate the passing of information using POST method. | 30 |
| 9. | Write a program to demonstrate the passing Checkbox Data to CGI Program | 31 |
| 10. | Write a program to demonstrate the passing Radio Button data to CGI program. | 32 |
| 11. | Write a program to demonstrate the passing Text Area data to CGI program. | 33 |

**Unit - I**

|  |
| --- |
| **Question 1** |
| Question: | | |
| Write a program which will find all such numbers which are divisible by 7 but are not a multiple of 5, between 2000 and 3200 (both included). | | |
| The numbers obtained should be printed in a comma-separated sequence on a single line. | | |
|  | | |
| Hints: | | |
| Consider use range(#begin, #end) method | | |
|  | | |
| Solution: | | |
| l=[] | | |
| for i in range(2000, 3201): | | |
| if (i%7==0) and (i%5!=0): | | |
| l.append(str(i)) | | |
|  | | |
| print ','.join(l) | | |
| #----------------------------------------# | | |
|  | | |
| #----------------------------------------# | |

|  |
| --- |
| **Question 2:** |
| With a given integral number n, write a program to generate a dictionary that contains (i, i\*i) such that is an integral number between 1 and n (both included) and then the program should print the dictionary. | | |
| Suppose the following input is supplied to the program: 8 | | |
| Then, the output should be: | | |
| {1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64} | | |
|  | | |
| Hints: | | |
| In case of input data being supplied to the question, it should be assumed to be a console input. | | |
| Consider use dict() | | |
|  | | |
| Solution: | | |
| n=int(raw\_input()) | | |
| d=dict() | | |
| for i in range(1,n+1): | | |
| d[i]=i\*i | | |
|  | | |
| print d | | |
| #----------------------------------------# | | |
|  | | |
| #----------------------------------------# | |

|  |
| --- |
| **Question 3:** |
| Write a program which accepts a sequence of comma-separated numbers from console and generate a list and a tuple which contains every number. | | |
| Suppose the following input is supplied to the program: | | |
| 34,67,55,33,12,98 | | |
| Then, the output should be: | | |
| ['34', '67', '55', '33', '12', '98'] | | |
| ('34', '67', '55', '33', '12', '98') | | |
|  | | |
| Hints: | | |
| In case of input data being supplied to the question, it should be assumed to be a console input. | | |
| tuple() method can convert list to tuple | | |
|  | | |
| Solution: | | |
| values=raw\_input() | | |
| l=values.split(",") | | |
| t=tuple(l) | | |
| print l | | |
| print t | | |
| #----------------------------------------# | | |
|  | | |
| #----------------------------------------# | |

|  |
| --- |
| **Question 4:** |
| Write a program that accepts a comma separated sequence of words as input and prints the words in a comma-separated sequence after sorting them alphabetically. | | |
| Suppose the following input is supplied to the program: | | |
| without,hello,bag,world | | |
| Then, the output should be: | | |
| bag,hello,without,world | | |
|  | | |
| Hints: | | |
| In case of input data being supplied to the question, it should be assumed to be a console input. | | |
|  | | |
| Solution: | | |
| items=[x for x in raw\_input().split(',')] | | |
| items.sort() | | |
| print ','.join(items) | | |
|  | | |
| #----------------------------------------# | | |
|  | |

**Question 5:**

Write a program that accepts sequence of lines as input and prints the lines after making all characters in the sentence capitalized.

|  |
| --- |
| Suppose the following input is supplied to the program: |
| Hello world |
| Practice makes perfect |
| Then, the output should be: |
| HELLO WORLD |
| PRACTICE MAKES PERFECT |
|  |
| Hints: |
| In case of input data being supplied to the question, it should be assumed to be a console input. |
|  |
| Solution: |
| lines = [] |
| while True: |
| s = raw\_input() |
| if s: |
| lines.append(s.upper()) |
| else: |
| break; |
|  |
| for sentence in lines: |
| print sentence |
|  |
|  |
|  |

**Question 6:**

|  |
| --- |
| Write a program that accepts a sentence and calculate the number of letters and digits. |
| Suppose the following input is supplied to the program: |
| hello world! 123 |
| Then, the output should be: |
| LETTERS 10 |
| DIGITS 3 |
|  |
| Hints: |
| In case of input data being supplied to the question, it should be assumed to be a console input. |
|  |
| Solution: |
| s = raw\_input() |
| d={"DIGITS":0, "LETTERS":0} |
| for c in s: |
| if c.isdigit(): |
| d["DIGITS"]+=1 |
| elif c.isalpha(): |
| d["LETTERS"]+=1 |
| else: |
| pass |
| print "LETTERS", d["LETTERS"] |
| print "DIGITS", d["DIGITS"] |
|  |
|  |
|  |

**Question 7:**

A website requires the users to input username and password to register. Write a program to check the validity of password input by users.

|  |
| --- |
| Following are the criteria for checking the password: |
| 1. At least 1 letter between [a-z] |
| 2. At least 1 number between [0-9] |
| 1. At least 1 letter between [A-Z] |
| 3. At least 1 character from [$#@] |
| 4. Minimum length of transaction password: 6 |
| 5. Maximum length of transaction password: 12 |
| Your program should accept a sequence of comma separated passwords and will check them according to the above criteria. Passwords that match the criteria are to be printed, each separated by a comma. |
| Example |
| If the following passwords are given as input to the program: |
| ABd1234@1,a F1#,2w3E\*,2We3345 |
| Then, the output of the program should be: |
| ABd1234@1 |
|  |
| Hints: |
| In case of input data being supplied to the question, it should be assumed to be a console input. |
|  |
| Solutions: |
| import re |
| value = [] |
| items=[x for x in raw\_input().split(',')] |
| for p in items: |
| if len(p)<6 or len(p)>12: |
| continue |
| else: |
| pass |
| if not re.search("[a-z]",p): |
| continue |
| elif not re.search("[0-9]",p): |
| continue |
| elif not re.search("[A-Z]",p): |
| continue |
| elif not re.search("[$#@]",p): |
| continue |
| elif re.search("\s",p): |
| continue |
| else: |
| pass |
| value.append(p) |
| print ",".join(value) |
|  |
|  |
|  |

**Question 8:**

Write a Program to demonstrate the reading and writing of text to a file.

#!/usr/bin/python

# Open a file

fo = open("foo.txt", "wb")

fo.write( "Python is a great language.\nYeah its great!!\n");

# Close opend file

fo.close()

#!/usr/bin/python

# Open a file

fo = open("foo.txt", "r+")

str = fo.read(10);

print "Read String is : ", str

# Close opend file

fo.close()

**UNIT-II**

|  |
| --- |
| 1. Write a program to find second largest and second smallest number in a given list. |
| def second\_largest(numbers): |
| count = 0 |
| n1 = n2 = float('-inf') |
| for x in numbers: |
| count += 1 |
| if x > n2: |
| if x >= n1: |
| n1, n2 = x, n1 |
| else: |
| n2 = x |
| return n2 if count >= 2 else None |
| print(second\_largest([1, 2, -8, -2, 0])) |
|  |
| def second\_smallest(numbers): |
| a1, a2 = float('inf'), float('inf') |
| for x in numbers: |
| if x <= a1: |
| a1, a2 = x, a1 |
| elif x < a2: |
| a2 = x |
| return a2 |
| print(second\_smallest([1, 2, -8, -2, 0])) |
|  |
| 2. Write a program to count the number of occurrence of characters (character frequency) in a string and number of occurrence of words in a sentence. |
| 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('google.com')) |
|  |
| def word\_count(str): |
| counts = dict() |
| words = str.split() |
| for word in words: |
| if word in counts: |
| counts[word] += 1 |
| else: |
| counts[word] = 1 |
| return counts |
| print( word\_count('the quick brown fox jumps over the lazy dog.')) |
| counts[word] += 1 |
|  |
| 3. Write a program to create a Caesar encryption. |
| Note: In cryptography, a Caesar cipher, also known as Caesar's cipher, the shift cipher, Caesar's code or Caesar shift, is one of the simplest and most widely known encryption techniques. It is a type of substitution cipher in which each letter in the plaintext is replaced by a letter some fixed number of positions down the alphabet. For example, with a left shift of 3, D would be replaced by A, E would become B, and so on. The method is named after Julius Caesar, who used it in his private correspondence. |
| def caesar\_encrypt(realText, step): |
| outText = [] |
| cryptText = [] |
| uppercase = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z'] |
| lowercase = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z'] |
| for eachLetter in realText: |
| if eachLetter in uppercase: |
| index = uppercase.index(eachLetter) |
| crypting = (index + step) % 26 |
| cryptText.append(crypting) |
| newLetter = uppercase[crypting] |
| elif eachLetter in lowercase: |
| index = lowercase.index(eachLetter) |
| crypting = (index + step) % 26 |
| cryptText.append(crypting) |
| newLetter = lowercase[crypting] |
| outText.append(newLetter) |
| return outText |
| code = caesar\_encrypt('abc', 2) |
| print() |
| print(code) |
| print() |
|  |
| 4. Write a program to create a list by entering countries and respective capital and population. The program should accept the name of a country as an input and print the corresponding capital name and population as output. Otherwise, the program should print an appropriate message if the country is not found in the list. |
| country = [] |
| ch = 'Y' |
| while ( ch == 'Y' or ch == 'YES'): |
| print ( ) |
| cname = input("enter country name:").upper() |
| capital = input("enter capital name:").upper() |
| c = ( population,cname,capital ) |
| country.append(c) |
| print ("do u want to enter more country<y/n>:") |
| ch = input().upper() |
| if ch =='N' or ch =='NO' : |
| break |
| cnm = input("enter the country name u wanna search").upper() |
| capname = " " |
| pop = 0 |
| flag = 0 |
| for c in country: |
| population,cname,capital = c |
| if(cnm == cname): |
| capname = capital |
| pop = population |
| flag = 1 |
| break |
| if ( flag == 1): |
| print("search country",cnm) |
| print ("capital",capname) |
| print ("population",pop) |
| else: |
| print ("sorry") |
| print(country) |

**UNIT-III**

|  |
| --- |
| 1. Write a binary search function which searches an item in a sorted list. The function should return the index of element to be searched in the list. |
| import math |
| defbin\_search(li, element): |
| bottom = 0 |
| top = len(li)-1 |
| index = -1 |
| while top>=bottom and index==-1: |
| mid = int(math.floor((top+bottom)/2.0)) |
| if li[mid]==element: |
| index = mid |
| elif li[mid]>element: |
| top = mid-1 |
| else: |
| bottom = mid+1 |
| return index |
| li=[2,5,7,9,11,17,222] |
| printbin\_search(li,l1) |
| printbin\_search(li,l2) |
|  |
| 2. Write a program to solve a classic ancient Chinese puzzle:  We count 35 heads and 94 legs among the chickens and rabbits in a farm. How many rabbits and how many chickens do we have? |
| def solve(numheads,numlegs): |
| ns='No solutions!' |
| for i in range(numheads+1): |
| j=numheads-i |
| if 2\*i+4\*j==numlegs: |
| returni,j |
| returnns,ns |
| numheads=35 |
| numlegs=94 |
| returni,j |
| returnns,ns |
| numheads=35 |
| numlegs=94 |
| solutions=solve(numheads,numlegs) |
| Print(solutions) |
|  |
| 3. With a given list [12, 24, 35, 24, 88, 120, 155, 88, 120, 155], write a program to print this list after removing all duplicate values with original order reserved. |
| defremoveDuplicate( li ): |
| newli=[] |
| seen = set() |
| for item in li: |
| if item not in seen: |
| seen.add( item ) |
| newli.append(item) |
| returnnewli |
| li=[12,24,35,24,88,120,155,88,120,155] |
| printremoveDuplicate(li) |
|  |
| 4. Write a program using list comprehension to print the Fibonacci Sequence in comma separated form with a given n input by console.  Example:  If the following n is given as input to the program:  7  Then, the output of the program should be:  0,1,1,2,3,5,8,13 |
| def f(n): |
| if n == 0: return 0 |
| elif n == 1: return 1 |
| else: return f(n-1)+f(n-2) |
| n=int(raw\_input()) |
| values = [str(f(x)) for x in range(0, n+1)] |
| print(“,”.join(values)) |
|  |
| 5. (Game: craps) Craps is a popular dice game played in casinos. Write a program to play a variation of the game, as follows:  Roll two dice. Each die has six faces representing values 1, 2, ..., and 6, respectively. Check the sum of the two dice. If the sum is 2, 3, or 12 (called craps), you lose; if the sum is 7 or 11 (called natural), you win; if the sum is another value (i.e., 4, 5, 6, 8, 9, or 10), a point is established. Continue to roll the dice until either a 7 or the same point value is rolled. If 7 is rolled, you lose. Otherwise, you win. Your program acts as a single player. |
| import random |
| import sys |
|  |
| def main(): |
| dice = getDice() |
| if dice == 7 or dice == 11: |
| print("You win") |
| sys.exit() |
| elif dice == 2 or dice == 3 or dice == 12: |
| print("You lose") |
| sys.exit() |
| point = dice |
| print("point is", point) |
| dice = getDice() |
| while dice != 7 and dice != point: |
| dice = getDice() |
| if dice == 7: |
| print("You lose") |
| else: |
| print("You win") |
| # Get a dice |
| defgetDice(): |
| i1 = random.randint(1, 6) |
| i2 = random.randint(1, 6) |
| print("You rolled", i1, "+", i2, "=", i1 + i2) |
| return i1 + i2 |
| main() |
|  |
| 6. Design a class named Quadratic Equation for a quadratic equation ax 2 + bx + x = 0. The class contains:  ■ The private data fields a , b , and c that represent three coefficients.  ■ A constructor for the arguments for a , b , and c .  ■ Three get methods for a , b , and c .  ■ A method named getDiscriminant() that returns the discriminant, which is b 2 - 4ac.  ■ The methods named getRoot1() and getRoot2() for returning the two roots of the equation  These methods are useful only if the discriminant is nonnegative. Let these methods return 0 if the discriminant is negative.  Write a test program that prompts the user to enter values for a, b, and c and displays the result based on the discriminant. If the discriminant is positive, display the two roots. If the discriminant is 0 , display the one root. Otherwise, display “The equation has no roots.” |
| import math |
|  |
| classQuadraticEquation: |
| def \_\_init\_\_(self, a, b, c): |
| self.\_\_a = a |
| self.\_\_b = b |
| self.\_\_c = c |
|  |
| defgetA(self): |
| returnself.\_\_a |
|  |
| defgetB(self): |
| returnself.\_\_b |
|  |
| defgetC(self): |
| returnself.\_\_c |
|  |
| defgetDiscriminant(self): |
| returnself.\_\_b \* self.\_\_b - 4 \* self.\_\_a \* self.\_\_c |
|  |
| def getRoot1(self): |
| if self.getDiscriminant() < 0: |
| return 0 |
| else: |
| return (-self.\_\_b + self.getDiscriminant()) / (2 \* self.\_\_a) |
|  |
| def getRoot2(self): |
| if self.getDiscriminant() < 0: |
| return 0 |
| else: |
| return (-self.\_\_b - self.getDiscriminant()) / (2 \* self.\_\_a) |
|  |
| def main(): |
| a, b, c = eval(input("Enter a, b, c: ")) |
| equation = QuadraticEquation(a, b, c) |
| discriminant = equation.getDiscriminant() |
|  |
| if discriminant < 0: |
| print("The equation has no roots") |
| elif discriminant == 0: |
| print("The root is", equation.getRoot1()) |
| else: # (discriminant >= 0) |
| print("The roots are", equation.getRoot1(), "and", equation.getRoot2()) |
| main() |
|  |
| 7. Design a class named Location for locating a maximal value and its location in a two-dimensional list. The class contains the public data fields row , column , and maxValue that store the maximal value and its indexes in a two-dimensional list, with row and column as int types and maxValue as a float type.  Write the following method that returns the location of the largest element in a two-dimensional list.  *def Location locateLargest(a):*  The return value is an instance of Location. Write a test program that prompts the user to enter a two dimensional list and displays the location of the largest element in the list. |
| class Location: |
| def \_\_init\_\_(self, row, column, maxValue): |
| self.row = row |
| self.column = column |
| self.maxValue = maxValue |
|  |
| defgetRow(self): |
| returnself.row |
|  |
| defgetColumn(self): |
| returnself.column |
|  |
| defgetMaxValue(self): |
| returnself.maxValue |
|  |
|  |
| def main(): |
| row, column = eval(input("Enter the number of rows and columns of the list: ")) |
| matrix = [] |
| for i in range(row): |
| s = input("Enter row " + str(i) + ": ") |
| items = s.split() # Extracts items from the string |
| list = [ eval(x) for x in items ] # Convert items to numbers |
| matrix.append(list) |
| location = locateLargest(matrix) |
| print("The location of the largest element is "  + str(location.getMaxValue()) + " at ("  + str(location.getRow()) + ", " + str(location.getColumn()) + ")") |
|  |
| deflocateLargest(a): |
| maxValue = a[0][0] |
| row = 0 |
| column = 0 |
| for i in range(len(a)): |
| for j in range(len(a[i])): |
| if maxValue< a[i][j]: |
| maxValue = a[i][j] |
| return Location(row, column, maxValue) |
| main() |
|  |
| 8. Demonstrate the working of Operator Overloading in python |
| import math |
| class Circle: |
| def \_\_init\_\_(self, radius): |
| self.\_\_radius = radius |
|  |
| defsetRadius(self, radius): |
| self.\_\_radius = radius |
|  |
| defgetRadius(self): |
| returnself.\_\_radius |
|  |
| def area(self): |
| returnmath.pi \* self.\_\_radius \*\* 2 |
|  |
| def \_\_add\_\_(self, another\_circle): |
| return Circle( self.\_\_radius + another\_circle.\_\_radius ) |
|  |
| def \_\_gt\_\_(self, another\_circle): |
| returnself.\_\_radius>another\_circle.\_\_radius |
|  |
| def \_\_lt\_\_(self, another\_circle): |
| returnself.\_\_radius<another\_circle.\_\_radius |
|  |
| def \_\_str\_\_(self): |
| return "Circle with radius " + str(self.\_\_radius) |
|  |
| c1 = Circle(4) |
| print(c1.getRadius()) |
| c2 = Circle(5) |
| print(c2.getRadius()) |
| c3 = c1 + c2 |
| print(c3.getRadius()) |
| print( c3 > c2) |
| print( c1 < c2) |
| print(c3) |

**UNIT-IV**

**4. GUI Programming using Tkinter**

Tkinter enables you to develop GUI programs and is an excellent pedagogical tool for learning object-oriented programming. The tkinter module contains the classes for creating GUIs. The Tk class creates a window for holding GUI widgets (i.e., visual components).

1. Write a program to demonstrate the creation of window, label and a button.

|  |
| --- |
| from tkinter import \* # Import all definitions from tkinter |
| window = Tk() # Create a window |
| label = Label(window, text = "Welcome to Python") # Create a label |
| button = Button(window, text = "Click Me") # Create a button |
| label.pack() # Place the label in the window |
| button.pack() # Place the button in the window |
| window.mainloop() # Create an event loop |

**4.1. Processing Events**

A Tkinter widget can be bound to a function, which is called when an event occurs.

2. Write a program to demonstrate the processing of the button event.

|  |
| --- |
| from tkinter import \* # Import all definitions from tkinter |
| def processOK(): |
| print("OK button is clicked") |
| def processCancel(): |
| print("Cancel button is clicked") |
| window = Tk() # Create a window |
| btOK = Button(window, text = "OK", fg = "red", command = processOK) |
| btCancel = Button(window, text = "Cancel", bg = "yellow", command = processCancel) |
| btOK.pack() # Place the OK button in the window |
| btCancel.pack() # Place the Cancel button in the window |
| window.mainloop() # Create an event loop |
| 3. Write a program that receives two numbers from the text fields and calculates the sum and then displays it in a message box. Generate an exception if the first number is less than the second. |
| from tkinter import \* |
| from InvalidNumberException import InvalidNumberException |
| class add: |
| def \_\_init\_\_(self): |
| window=Tk() |
| window.title("Addition of two numbers") |
| frame1=Frame(window) |
| frame1.pack() |
| label1=Label(frame1,text="First Number") |
| label1.grid(row=1,column=1) |
| self.v1=IntVar() |
| entry1=Entry(frame1,textvariable=self.v1) |
| entry1.grid(row=1,column=2) |
| label2=Label(frame1 |
| label2.grid(row=2 |
| self.v2=IntVar() |
| entry2=Entry(frame1,textvariable=self.v2) |
| entry2.grid(row=2,column=2) |
| bt=Button(frame1,text="Add",command=self.processButton) |
| bt.grid(row=3,column=2) |
| self.message=Message(frame1,text="") |
| self.message.grid(row=4,column=1,columnspan=4) |
| window.mainloop() |
| def processButton(self): |
| first=self.v1.get() |
| second=self.v2.get() |
| if self.v1.get() > self.v2.get(): |
| self.v3=self.v1.get() + self.v2.get() |
| self.message["text"]="The sum is:"+str(self.v3) |
| else: |
| raise InvalidNumberException(first,second) |
| add() |
|  |
| 4. Create a GUI that is having a text box to enter the name, radio button to select the gender, check-box to choose hobbies. Write a function that’s gets called on the button click to display these values as labels. |
| from tkinter import \* |
| class details: |
| def \_\_init\_\_(self): |
| window=Tk() |
| window.title("Person details") |
| self.frame1=Frame(window) |
| self.frame1.pack() |
| self.label1=Label(self.frame1,text="Name:") |
| self.label1.grid(row=1,column=1) |
| self.name=StringVar() |
| entry1=Entry(self.frame1,textvariable=self.name) |
| entry1.grid(row=1,column=2) |
| self.label2=Label(self.frame1,text="Gender:") |
| self.label2.grid(row=2,column=1) |
| self.v1=IntVar() |
| rbMale=Radiobutton(self.frame1,text="Male",variable=self.v1,value=1,  command=self.processRadiobutton) |
| rbMale.grid(row=2,column=2) |
| rbFemale=Radiobutton(self.frame1,text="Female",variable=self.v1,value=2  ,command=self.processRadiobutton) |
| rbFemale.grid(row=2,column=3) |
| self.label3=Label(self.frame1,text="Hobbies:") |
| self.label3.grid(row=3,column=1) |
| self.v2=IntVar() |
| cb1=Checkbutton(self.frame1,text="Painting",variable=self.v2,  command=self.processCheckbutton) |
| self.v3=IntVar() |
| cb2=Checkbutton(self.frame1,text="Hiking",variable=self.v3,  command=self.processCheckbutton) |
| self.v4=IntVar() |
| cb3=Checkbutton(self.frame1,text="Paragliding",variable=self.v4,  command=self.processCheckbutton) |
| cb1.grid(row=3,column=2) |
| cb2.grid(row=3,column=3) |
| cb3.grid(row=3,column=4) |
| bt=Button(self.frame1,text="Display details",command=self.processButton) |
| bt.grid(row=4 ,column=2) |
| window.mainloop() |
|  |
| def processRadiobutton(self): |
| if self.v1.get()== 1: |
| print("Male option is checked") |
| else: |
| print("Female option is checked") |
|  |
| def processCheckbutton(self): |
| if self.v2.get()==1 and self.v3.get()==1 and self.v4.get()==1: |
| print("Painting, hiking and paragliding option is clicked") |
| elif self.v2.get()==1 and self.v3.get()==1: |
| print("Painting and Hiking option is clicked") |
| elif self.v3.get()==1 and self.v4.get()==1: |
| print("Hiking and Paragliding option is clicked") |
| elif self.v2.get()==1 and self.v4.get()==1: |
| print("Painting and Paragliding option is clicked") |
| elif self.v2.get()==1: |
| print("Painting option is checked") |
| elif self.v3.get()==1: |
| print("Hiking option is checked") |
| elif self.v4.get()==1: |
| print("Paragliding option is checked") |
| def processButton(self): |
| self.label4=Label(self.frame1,text="Name:" +self.name.get()) |
| self.label4.grid(row=5,column=1) |
| if self.v2.get()==1: |
| self.label5=Label(self.frame1,text="Gender:Male") |
| self.label5.grid(row=6,column=1) |
| else: |
| self.label5=Label(self.frame1,text="Gender:Female") |
| self.label5.grid(row=6,column=1) |
| if self.v2.get()==1 and self.v3.get()==1 and self.v4.get()==1: |
| self.label6=Label(self.frame1,text="Hobbies:Painting, Hiking and Paragliding") |
| self.label6.grid(row=7,column=1) |
| elif self.v2.get()==1 and self.v3.get()==1: |
| self.label7=Label(self.frame1,text="Hobbies:Painting and Hiking") |
| self.label7.grid(row=7,column=1) |
| elif self.v3.get()==1 and self.v4.get()==1: |
| self.label8=Label(self.frame1,text="Hobbies:Hiking and Paragliding") |
| self.label8.grid(row=7,column=1) |
| elif self.v2.get()==1 and self.v4.get()==1: |
| self.label9=Label(self.frame1,text="Hobbies:Painting and Paragliding") |
| self.label9.grid(row=7,column=1) |
| elif self.v2.get()==1: |
| self.label10=Label(self.frame1,text="Hobby:Painting") |
| self.label10.grid(row=7,column=1) |
| elif self.v3.get()==1: |
| self.label11=Label(self.frame1,text="Hobby:Hiking") |
| self.label11.grid(row=7,column=1) |
| elif self.v4.get()==1: |
| self.label12=Label(self.frame1,text="Hobby:Paragliding") |
| self.label12.grid(row=7,column=1) |
| details() |
|  |
| 5. Write a program to demonstrate the concept of multithreading. |
| import threading |
| import time |
| exitFlag = 0 |
| class myThread (threading.Thread): |
| def \_\_init\_\_(self, threadID, name, counter): |
| threading.Thread.\_\_init\_\_(self) |
| self.threadID = threadID |
| self.name = name |
| self.counter = counter |
| def run(self):  print ("Starting " + self.name)  print\_time(self.name, self.counter, 5)  print ("Exiting " + self.name) |
| def print\_time(threadName, delay, counter):  while counter:  if exitFlag:  threadName.exit()  time.sleep(delay)  print ("%s: %s" % (threadName, time.ctime(time.time())))  counter -= 1  # Create new threads  thread1 = myThread(1, "Thread-1", 1)  thread2 = myThread(2, "Thread-2", 2)  # Start new Threads  thread1.start()  thread2.start()  thread1.join()  thread2.join()  print ("Exiting Main Thread") |

6. Write a program to demonstrate the concept of synchronizing the threads

import threading

import time

class myThread (threading.Thread):

def \_\_init\_\_(self, threadID, name, counter):

threading.Thread.\_\_init\_\_(self)

self.threadID = threadID

self.name = name

self.counter = counter

def run(self):

print ("Starting " + self.name)

# Get lock to synchronize threads

threadLock.acquire()

print\_time(self.name, self.counter, 3)

# Free lock to release next thread

threadLock.release()

def print\_time(threadName, delay, counter):

while counter:

time.sleep(delay)

print ("%s: %s" % (threadName, time.ctime(time.time())))

counter -= 1

threadLock = threading.Lock()

threads = []

# Create new threads

thread1 = myThread(1, "Thread-1", 1)

thread2 = myThread(2, "Thread-2", 2)

# Start new Threads

thread1.start()

thread2.start()

# Add threads to thread list

threads.append(thread1)

threads.append(thread2)

# Wait for all threads to complete

for t in threads:

t.join()

print ("Exiting Main Thread")

7. Write a program to demonstrate the concept of client server program.

**Server Program**

import socket

import os.path

# create a socket object

serversocket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

# get local machine name

host = socket.gethostname()

print("Host name:",host)

port = 9999

# bind to the port

serversocket.bind((host,port))

# queue up to 5 requests

serversocket.listen(5)

while True:

# establish a connection

clientsocket,addr=serversocket.accept()

print("Got a connection from %s" % str(addr))

f=clientsocket.recv(1024)

if os.path.isfile(f):

infile=open(f,"r")

for line in infile:

clientsocket.send(line.encode('ascii'))

msg='Thank you for connecting'+ "\r\n"

clientsocket.send(msg.encode('ascii'))

else:

f1="File doesn't exist"

clientsocket.send(f1.encode('ascii'))

clientsocket.close()

**Client Program**

import socket

# create a socket object

s = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

# get local machine name

host = socket.gethostname()

print("Host name:",host)

port = 9999

# connection to hostname on the port.

s.connect((host,port))

f1=input("Enter the file name:")

s.send(f1.encode('ascii'))

# Receive no more than 1024 bytes

msg = s.recv(1024)

s.close()

print (msg.decode('ascii'))

**UNIT-V**

5.1. The Python standard for database interfaces is the Python DB-API. Most Python database interfaces adhere to this standard.

You can choose the right database for your application. Python Database API supports a wide range of database servers such as −

* GadFly
* mSQL
* MySQL
* PostgreSQL
* Microsoft SQL Server 2000
* Informix
* Interbase
* Oracle
* Sybase
* SQLite

The DB API provides a minimal standard for working with databases using Python structures and syntax wherever possible. This API includes the following:

* Importing the API module.
* Acquiring a connection with the database.
* Issuing SQL statements and stored procedures.
* Closing the connection

**5.2. PyMySQL**

PyMySQL is an interface for connecting to a MySQL database server from Python. It implements the Python Database API v2.0 and contains a pure-Python MySQL client library.

**5.2.1 Installation of PyMySQL**

Before proceeding furthur, you make sure you have PyMySQL installed on your machine. Just type the following in your Python script and execute it –

#!/usr/bin/python3

import PyMySQL

If it produces the following result, then it means MySQLdb module is not installed:

Traceback (most recent call last):

File "test.py", line 3, in <module>

Import PyMySQL

ImportError: No module named PyMySQL

The last stable release is available on PyPI and can be installed with pip −

pip install PyMySQL

**5.3. Database Connection**

Before connecting to a MySQL database, make sure of the following points −

* You have created a database TESTDB.
* You have created a table EMPLOYEE in TESTDB.
* This table has fields FIRST\_NAME, LAST\_NAME, AGE, SEX and INCOME.
* User ID "testuser" and password "test123" are set to access TESTDB.
* Python module PyMySQL is installed properly on your machine.
* You have gone through MySQL tutorial to understand [MySQL Basics.](https://www.tutorialspoint.com/mysql/index.htm)

1. Write a program to create a database table EMPLOYEE.

import PyMySQL

# Open database connection

db = PyMySQL.connect("localhost","testuser","test123","TESTDB" )

# prepare a cursor object using *cursor()* method

cursor = db.cursor()

# Drop table if it already exist using *execute()* method.

cursor.execute("DROP TABLE IF EXISTS EMPLOYEE")

# Create table as per requirement

sql = """CREATE TABLE EMPLOYEE (

FIRST\_NAME CHAR(20) NOT NULL,

LAST\_NAME CHAR(20),

AGE INT,

SEX CHAR(1),

INCOME FLOAT )"""

cursor.execute(sql)

# disconnect from server

db.close()

2. Write a program which executes SQL *INSERT* statement to create a record in the EMPLOYEE table

import PyMySQL

# Open database connection

db = PyMySQL.connect("localhost","testuser","test123","TESTDB" )

# prepare a cursor object using cursor() method

cursor = db.cursor()

# Prepare SQL query to INSERT a record into the database.

sql = """INSERT INTO EMPLOYEE(FIRST\_NAME,

LAST\_NAME, AGE, SEX, INCOME)

VALUES ('Mac', 'Mohan', 20, 'M', 2000)"""

try:

# Execute the SQL command

cursor.execute(sql)

# Commit your changes in the database

db.commit()

except:

# Rollback in case there is any error

db.rollback()

# disconnect from server

db.close()

3. Write a program that updates all the records having SEX as **'M'**. Here, the AGE of all the males is increased by one year.

import PyMySQL

# Open database connection

db = PyMySQL.connect("localhost","testuser","test123","TESTDB" )

# prepare a cursor object using *cursor()* method

cursor = db.cursor()

# Prepare SQL query to UPDATE required records

sql = "UPDATE EMPLOYEE SET AGE = AGE + 1

WHERE SEX = '%c'" % ('M')

try:

# Execute the SQL command

cursor.execute(sql)

# Commit your changes in the database

db.commit()

except:

# Rollback in case there is any error

db.rollback()

# disconnect from server

db.close()

4. DELETE operation is required when you want to delete some records from your database. Following is the program to delete all the records from EMPLOYEE where AGE is more than 20.

import PyMySQL

# Open database connection

db = PyMySQL.connect("localhost","testuser","test123","TESTDB" )

# prepare a cursor object using *cursor()* method

cursor = db.cursor()

# Prepare SQL query to DELETE required records

sql = "DELETE FROM EMPLOYEE WHERE AGE > '%d'" % (20)

try:

# Execute the SQL command

cursor.execute(sql)

# Commit your changes in the database

db.commit()

except:

# Rollback in case there is any error

db.rollback()

# disconnect from server

db.close()

**5.4. CGI Programming**

The Common Gateway Interface, or CGI, is a standard for external gateway programs to interface with information servers such as HTTP servers.

5. Write sample program to demonstrate the concept of CGI.

The program is written with the name of the script as hello.py. This file is kept in /var/www/cgi-bin directory and it has following content.

#!/usr/bin/python3

print ("Content-type:text/html")

print()

print ("<html>")

print ('<head>')

print ('<title>Hello Word - First CGI Program</title>')

print ('</head>')

print ('<body>')

print ('<h2>Hello Word! This is my first CGI program</h2>')

print ('</body>')

print ('</html>')

Enter following URL in the browser:

<http://localhost:8080/cgi-bin/hello.py>

Output:

Hello Word! This is my first CGI program

**6. Passing Information using GET method**

Below is **hello\_get.py** script to handle input given by web browser. We are going to use **cgi** module, which makes it very easy to access passed information.

# Import modules for CGI handling

import cgi, cgitb

# Create instance of FieldStorage

form = cgi.FieldStorage()

# Get data from fields

first\_name = form.getvalue('first\_name')

last\_name = form.getvalue('last\_name')

print ("Content-type:text/html")

print()

print ("<html>)"

print ("<head>")

print ("<title>Hello - Second CGI Program</title>")

print ("</head>")

print ("<body>")

print ("<h2>Hello %s %s</h2>" % (first\_name, last\_name))

print ("</body>")

print ("</html>">)

**7. Simple FORM Example:GET Method**

This example passes two values using HTML FORM and submit button. We use same CGI script hello\_get.py to handle this input.

<form action="/cgi-bin/hello\_get.py" method="get">

First Name: <input type="text" name="first\_name"> <br />

Last Name: <input type="text" name="last\_name" />

<input type="submit" value="Submit" />

</form>

**8. Passing Information Using POST Method**

Below is same hello\_get.py script which handles GET as well as POST method.

# Import modules for CGI handling

import cgi, cgitb

# Create instance of FieldStorage

form = cgi.FieldStorage()

# Get data from fields

first\_name = form.getvalue('first\_name')

last\_name = form.getvalue('last\_name')

print ("Content-type:text/html")

print()

print ("<html>")

print ("<head>")

print ("<title>Hello - Second CGI Program</title>")

print ("</head>")

print ("<body>")

print ("<h2>Hello %s %s</h2>" % (first\_name, last\_name))

print ("</body>")

print ("</html>")

Let us take again same example as above which passes two values using HTML FORM and submit button.

<form action="/cgi-bin/hello\_get.py" method="post">

First Name: <input type="text" name="first\_name"><br />

Last Name: <input type="text" name="last\_name" />

<input type="submit" value="Submit" />

</form>

## 9. Passing Checkbox Data to CGI Program

Here is example HTML code for a form with two checkboxes.

<form action="/cgi-bin/checkbox.py" method="POST" target="\_blank">

<input type="checkbox" name="maths" value="on" /> Maths

<input type="checkbox" name="physics" value="on" /> Physics

<input type="submit" value="Select Subject" />

</form>

Below is checkbox.cgi script to handle input given by web browser for checkbox button.

# Import modules for CGI handling

import cgi, cgitb

# Create instance of FieldStorage

form = cgi.FieldStorage()

# Get data from fields

if form.getvalue('maths'):

math\_flag = "ON"

else:

math\_flag = "OFF"

if form.getvalue('physics'):

physics\_flag = "ON"

else:

physics\_flag = "OFF"

print ("Content-type:text/html")

print()

print ("<html>")

print ("<head>")

print ("<title>Checkbox - Third CGI Program</title>")

print ("</head>")

print ("<body>")

print ("<h2> CheckBox Maths is : %s</h2>" % math\_flag)

print ("<h2> CheckBox Physics is : %s</h2>" % physics\_flag)

print ("</body>")

print ("</html>")

**10. Passing Radio Button Data to CGI Program**

Here is example HTML code for a form with two radio buttons.

<form action="/cgi-bin/radiobutton.py" method="post" target="\_blank">

<input type="radio" name="subject" value="maths" /> Maths

<input type="radio" name="subject" value="physics" /> Physics

<input type="submit" value="Select Subject" />

</form>

Below is radiobutton.py script to handle input given by web browser for radio button:

# Import modules for CGI handling

import cgi, cgitb

# Create instance of FieldStorage

form = cgi.FieldStorage()

# Get data from fields

if form.getvalue('subject'):

subject = form.getvalue('subject')

else:

subject = "Not set"

print "Content-type:text/html")

print()

print ("<html>")

print ("<head>")

print ("<title>Radio - Fourth CGI Program</title>")

print ("</head>")

print ("<body>")

print ("<h2> Selected Subject is %s</h2>" % subject)

print ("</body>")

print ("</html>")

11. **Passing Text Area Data to CGI Program**

Here is example HTML code for a form with a TEXTAREA box

<form action="/cgi-bin/textarea.py" method="post" target="\_blank">

<textarea name="textcontent" cols="40" rows="4">

Type your text here...

</textarea>

<input type="submit" value="Submit" />

</form>

Below is textarea.cgi script to handle input given by web browser

# Import modules for CGI handling

import cgi, cgitb

# Create instance of FieldStorage

form = cgi.FieldStorage()

# Get data from fields

if form.getvalue('textcontent'):

text\_content = form.getvalue('textcontent')

else:

text\_content = "Not entered"

print "Content-type:text/html")

print()

print ("<html>")

print ("<head>";)

print ("<title>Text Area - Fifth CGI Program</title>")

print ("</head>")

print ("<body>")

print ("<h2> Entered Text Content is %s</h2>" % text\_content)

print ("</body>")

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*