

**Department of Electronics and Communication Engineering**

**Course Code and Name : U23CS382/Python Programming**

**Programme : B.E**

**Year/Semester : I / II Semester**

|  |  |  |  |
| --- | --- | --- | --- |
| **MODULE 1- PYTHON CONSTRUCTS** | | | |
| **PART A (20 QUESTIONS)** | | | |
| **S.NO** | **QUESTIONS** | **BT LEVEL** | **COGNIZANCE LEVEL** |
| 1. | **Comment with an example on the use of local and global variable with the same identifier name.**  The scope of a variable refers to the places that you can see or access a variable. If we define a variable on the top of the script or module, the variable is called global variable. The variables that are defined inside a class or function is called local variable.  **Example:**  def my\_local():  a=10  print(“This is local variable”)  Example:  a=10  def my\_global():  print(“This is global variable”) | **K1** | **Remember** |
| 2. | **Compare string and string slices.**  A string is a sequence of character.  **Eg:** fruit = ‘banana’  **String Slices :**  A segment of a string is called string slice, selecting a slice is similar to selecting a character.  **Eg:**  >>> s ='Monty Python'  >>> print s[0:5]  Monty  >>> print s[6:12]  Python | **K1** | **Remember** |
| 3. | **Distinguish between script mode and interactive mode**   |  |  | | --- | --- | | **Interactive Mode** | **Script Mode** | | A way of using the Python interpreter by typing commands and expressions at the prompt | A way of using the Python interpreter to read and execute statements in a script | | Can’t save and edit the code | Can save and edit the code | | We can see the results immediately | We cannot see the results immediately | | **K2** | Understand |
| 4. | **Predict the output of the following nested loop?**  **for num in range(10, 14):**  **for i in range(2, num):**  **if num%i == 1:**  **print(num)**  **break**  Output:  11  12  13 | **K4** | **Analyze** |
| 5. | **Write a python code to display the digit’s at one’s place of a number**  a=**int**(input("Enter any number :\n"))  ones\_place=a%10  **print**("The digits at one\'s place is %d"%ones place) | **K3** | **Apply** |
| 6. | **Generate a python program to print whether a number is positive or negative.**  n=int(input("Enter number: "))  **if**(n>0):  **print**("Number is positive")  else:  **print**("Number is negative") | **K3** | **Apply** |
| 7. | **Write a Python program to print the numbers in a range which are not divisible by 2 and 3 for** i **in** range(0,51):  **if**(i%2!=0&i%3!=0):  **print**(i) | **K3** | **Apply** |
| 8. | **Write a program to  swap two numbers without using temporary variable**  x = 5  y = 10  x, y = y, x  **print**("x =", x)  **print**("y =", y) | **K4** | **Analyze** |
| 9. | **Write a Python program to find the remainder of a/b, without using % operator.**  x=**int**(input("Enter the value of a :\n"))  y=**int**(input("Enter the value of b :\n"))  q=x//y  r=x-(y\*q);  **print**("Reminder is",r) | **K4** | **Analyze** |
| 10. | **Write a python program to print whether a number is positive or negative.**  n=int(input("Enter number: "))  **if**(n>0):  **print**("Number is positive")  else:  **print**("Number is negative") | **K2** | **Understand** |
| 11. | **Write a Python program to reverse a given number** n=int(input("Enter number: "))  rev=0  **while**(n>0):  dig=n%10  rev=rev\*10+dig  n=n//10  **print**("Reverse of the number:",rev) | **K3** | **Apply** |
| 12. | **Write a Python program to print the numbers in a range which are not divisible by 2 and 3 for** i **in** range(0,51):  **if**(i%2!=0&i%3!=0):  **print**(i) | **K4** | **Analyze** |
| 13. | **Differentiate Compiler and Interpreter**   |  |  | | --- | --- | | **Compiler** | **Interpreter** | | Scans the entire program and translates it as a whole into machine code | Translates one statement at a time | | It takes large amount of time to analyze the source code but the overall execution time is comparatively faster | It takes less amount of time to analyze the source code but the overall execution time is slower | | Generates intermediate object code which further requires linking, hence requires more memory | No intermediate object code is generated, hence memory is efficient | | **K2** | **Understand** |
| 14. | **Mention a few string functions.**  s.captilize() – Capitalizes first character of string  s.count(sub) – Count number of occurrences of sub in string  s.lower() – converts a string to lower case  s.split() – returns  a list of words in string | **K1** | **Remember** |
| 15. | **Write a Python program to find the remainder of a/b, without using % operator.**  x=**int**(input("Enter the value of a :\n"))  y=**int**(input("Enter the value of b :\n"))  q=x//y  r=x-(y\*q);  **print**("Reminder is",r) | **K4** | **Analyze** |
| 16. | **Write a python program to find the area of a circle.** radius=**int**(input())  pi=3.14 area=pi\*radius\*radius  **print**('%.2f'%area) | **K3** | **Apply** |
| 17. | **Write a python program for calculating Simple Interest**  P=**int**(input("Enter the principal amount"))  N=**float**(input("\nEnter the rate of interest"))  R=**int**(input("\nEnter the time period (in years)"))  SI=(P\*N\*R)/100  **print**('\nSimple Interest is %.2f'%SI) | **K3** | **Apply** |
| 18. | **Write a python program for concatenating two strings.**  string1=**input**("Enter string1:")  string2=**input**("Enter string2:")  string3=string1+string2  **print**('The concatenated string is %s'%string3) | **K3** | **Apply** |
| 19. | |  |  | | --- | --- | | **Write a Python Program to count the occurrences of the substring in a given string** |  | | string = **input**("Enter the string\n")  substring = **input**("Enter the sub string:")  count = **string.count**(substring)  **print**("The count is:", count) | | | **K4** | **Analyze** |
| 20. | **Write a python program to print ascii value of a character.**  c = 'g'  **print**("The ASCII value of '" + c + "' is", ord(c)) | **K3** | **Apply** |
| **PART B ( 7 QUESTIONS)** | | | |
| 1. | **There are n kids with candies. You are given an integer array candies, where each candies[i] represents the number of candies the ith kid has, and an integer extra Candies, denoting the number of extra candies that you have.**  **Return a boolean array result of length n, where result[i] is true if, after giving the ith kid all the extraCandies, they will have the greatest number of candies among all the kids, or false otherwise.**  **Note that multiple kids can have the greatest number of candies.[Leetcode]**  **Solution**  def kidsWithCandies(candies, extraCandies):  # Find the maximum number of candies among all kids  max\_candies = max(candies)    # Check if each kid can have the greatest number of candies  result = [candy + extraCandies >= max\_candies for candy in candies]    return result  # Example usage  candies = [2, 3, 5, 1, 3]  extraCandies = 3  print(kidsWithCandies(candies, extraCandies)) # Output: [True, True, True, False, True] | **K4** | **Analyze** |
| 2. | **Explain about the data types in python with suitable example**  The standard data types are   * Integer Type * Floating Point Type * String Type * Boolean Type * List Type   **Integer Type**   * Integers are **whole numbers** with no fractional part and decimal point. * They can be either positive, negative or zero value. * To write an integer in decimal (base 10), the first digit must not be zero * **Example**   x = 5  **print**(type(x))  **Floating Point Type**   * A floating point (float) type represents numbers with fractional part. * A floating point number has a decimal point and a fractional part. * Alternatively, floats may be expressed in scientific notation using letter “e” to indicate 10th power. * **Example**   a\_float = 3.14159  formatted\_float = "{:.2f}".format(a\_float)  **print**(formatted\_float)  **String Type**   * A string represents sequence of characters * It can be created using Single Quotes, Double Quotes and triple quotes * **Example** * Using Single Quotes: ‘HELLO’ * Using Double Quotes: “HELLO” * Using Triple Quotes : ‘‘‘ Hello Every One   Welcome to Python Programming’’’  **Boolean Type**   * A Boolean type represents special values ‘True’ and ‘False’ * They are represented as 1 and 0 * The most common way to produce a Boolean value is with a relational operator * **Example**:   2<3 is True  **List Type**   * List is an **ordered sequence of items** * Values in the list are called **elements/items** * Lists are created by **placing all items inside a square bracket separated by commas** * Items in a list can be of different data type * Lists are **mutable** * **Example**   my\_list = ['Book', 'Pen', 'Pencil']  **print**(my\_list[0], my\_list[2]) | **K2** | **Understand** |
| 3. | **Appraise the various expressions in python with an example**   * An expression is a combination of operators and operands that is interpreted to produce some other value. * An expression is evaluated as per the precedence of its operators. The expression types are * Constant Expressions * Arithmetic Expressions * Integral Expressions * Floating Expressions * Relational Expressions * Logical Expressions * Bitwise Expressions   **Constant Expressions**  Constant expressions are expressions having constant values only.  **Example**  x = 15 + 1.3  print(x)  **Output**  16.3  **Arithmetic Expression**  An arithmetic expression is a combination of numeric values, operators, and sometimes parenthesis.The arithmetic operators are    **Integral Expressions**  These are the kind of expressions that produce only **integer results** after all computations and type conversions.  **Program**  a = 13  b = 12.0  c = a + int(b)  print(c)  **Output**  25  **Floating Point Expressions**  These are the kind of expressions which produce floating point numbers as result after all computations and type conversions.  **Example:**  a = 13  b = 5    c = a / b  print(c)  **Output**  2.6  **Relational Expressions**  These expressions compare the operand values in both sides. The relational operators in python return a boolean value, i.e., either True or False based on the value of operands.  **Example:**  # Relational Expressions  a = 21  b = 13  c = 40  d = 37  p = (a + b) >= (c - d)  print(p)  **Output**  True  **Logical Expressions**  These are kinds of expressions that result in either True or False. It basically specifies one or more conditions.  For example, (10 == 9) is a condition if 10 is equal to 9 and will return False.    **Bitwise Expressions**  These are the kind of expressions in which computations are performed at bit level  **Example** a = 12  x = a >> 2  y = a << 1  print(x, y)  **Output**  3 24 | **K2** | **Understand** |
| 4. | **a) Write a Python program to find the factorial of the given number without recursion with recursion.**  n=int(input("Enter number:"))  fact=1  while(n>0):      fact=fact\*n      n=n-1  print("Factorial of the number is: ")  print(fact)  **b)Write a Python program to generate first ‘N’ Fibonacci series numbers.(Note: Fibonacci numbers are 0, 1,1,2,3,5,8… where each number is the sum of the preceding two).**  def Fibonacci(n):        # Check if input is 0 then it will      # print incorrect input      if n < 0:          print("Incorrect input")        # Check if n is 0      # then it will return 0      elif n == 0:          return 0        # Check if n is 1,2      # it will return 1      elif n == 1 or n == 2:          return 1        else:          return Fibonacci(n-1) + Fibonacci(n-2)    # Driver Program  print(Fibonacci(9)) | **K4** | **Analyze** |
| 5. | **Write a python code for developing a simple financial application** | **K4** | **Analyze** |
| 6. | **a)Write a Python program to find the factorial of a number provided by the user.**  # change the value for a different result  num = 7  # To take input from the user  #num = **int(input**("Enter a number: "))  factorial = 1  # check if the number is negative, positive or zero  **if** num < 0:  **print**("Sorry, factorial does not exist for negative numbers")  **elif** num == 0:  **print**("The factorial of 0 is 1")  **else:**  **for** i in range(1,num + 1):  factorial = factorial\*i  **print**("The factorial of",num,"is",factorial)  **b)Write a program to add the digits of a given number.**  **print**(end="Enter a Number: ")  num = **int**(input())  sum = 0  **print**(end="\n")  **while** num>0:  rem = num%10  sum = sum+rem  num = int(num/10)  **if** num==0:  **print**(end=str(rem))  **else:**  **print**(end=str(rem)+ "+")  **print**(" = " +str(sum)) | **K4** | **Analyze** |
| 7. | **a) Write a program to check whether a given number is palindrome or not**  n=**int(input**("Enter number:"))  temp=n  rev=0  **while**(n>0):  dig=n%10  rev=rev\*10+dig  n=n//10  **if**(temp==rev):  **print**("The number is a palindrome!")  **else:**  **print**("The number isn't a palindrome!")  **b)Write a program to check whether a given number is perfect number or not**  **print**("Enter the Number:")  num = int(input())  sum = 0  **for** i in range(1, num):  **if** num%i==0:  sum = sum+i  **if** num==sum:  **print**("It is a Perfect Number")  **else:**  **print**("It is not a Perfect Number") | **K4** | **Analyze** |

**Faculty Incharge HoD**