**Computer Practical**

Python: Grade 12

*Rayhan Mohammad Riyaz*

*12 A*

**Contents**

|  |  |  |  |
| --- | --- | --- | --- |
| Chapter No. | Chapter Name | Page No. | Teacher’s Signature |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Chapter 1: Python Revision Tour

## Solved Problems (pg 36)

|  |  |
| --- | --- |
| 22 | **Question:** Write a program that asks the user to input number of seconds and then expresses it in terms of many minutes and seconds it contains. |
| **Code:**  total\_seconds = int(input("Number of Seconds: "))  minutes = total\_seconds//60  seconds = total\_seconds%60  print(total\_seconds, "is", minutes, "minutes and", seconds, "seconds.") |
| **Output:**  Number of Seconds: 393  393 is 6 minutes and 33 seconds. |
| 23 | **Question:** Write a program that repeatedly asks from users some numbers until string ‘done’ is typed. The program should print the sum of all numbers entered. |
| **Code:**  num\_sum = 0  while True:      num = input("Number to be added (or 'done' to stop): ")      if num=='done':          break      else:          num = int(num)      num\_sum += num  print("Sum of the numbers inputted:", num\_sum) |
| **Output:**  Number to be added (or 'done' to stop): 13  Number to be added (or 'done' to stop): 2  Number to be added (or 'done' to stop): -5  Number to be added (or 'done' to stop): done  Sum of the numbers inputted: 10 |
| 24 | **Question:** Write a program to print a square multiplication table as shown below |
| **Code:**  hor = input("Horizontal Width of Multiplication Table (default=10): ")  ver = input("Vertical Height of Multiplication Table (default=10): ")  if not hor:      hor=10  if not ver:      ver=10  max\_spaces = len(str(hor\*ver))  for i in range(1, ver+1):      for j in range(i, hor\*i+1, i):          print(j, end=' '\*(max\_spaces-len(str(j))))      print() |
| **Output:**  Horizontal Width of Multiplication Table (default=10):  Vertical Height of Multiplication Table (default=10):  1 2 3 4 5 6 7 8 9 10  2 4 6 8 10 12 14 16 18 20  3 6 9 12 15 18 21 24 27 30  4 8 12 16 20 24 28 32 36 40  5 10 15 20 25 30 35 40 45 50  6 12 18 24 30 36 42 48 54 60  7 14 21 28 35 42 49 56 63 70  8 16 24 32 40 48 56 64 72 80  9 18 27 36 45 54 63 72 81 90  10 20 30 40 50 60 70 80 90 100 |

## Type C

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | **Question:** Write a program to print one of the words negative, zero or positive, according to whether variable x is less than zero, zero or greater than zero, respectively. | | | |
| **Code:**  x = int(input("Number: "))  print(x, "is", end=" ")  if x<0:      print("negative (less than 0).")  elif x==0:      print("zero (0).")  else:      print("positive (greater than 0).") | | | |
| **Output:** | | | |
| Number: 7  7 is positive (greater than 0). | Number: 0  0 is zero (0). | | Number: -3  -3 is negative (less than 0). |
| 3 | **Question:** Write a Python program that calculates and prints the number of seconds in a year | | | |
| **Code:**  num\_years = 1  num\_days = num\_years\*365  num\_hours = num\_days\*24  num\_minutes = num\_hours\*60  num\_seconds = num\_minutes\*60  print("Number of seconds in", num\_years, "years is", num\_seconds) | | | |
| **Output:**  Number of seconds in 1 years is 31536000 | | | |
| 5 | **Question:** Write a program that asks the user the day number in a year in the range 2 to 365 and asks the first day of the year – Sunday, Monday, Tuesday, etc. Then the program should display the day on the day-number that has been inputted. | | | |
| **Code:**  first\_day\_str = input("First day of the year (in lowercase): ")  date = int(input("Which day number of the year do you want to know the day of (2 to 365): "))  first\_day=0  if first\_day\_str=="monday":  first\_day=1  elif first\_day\_str=="tuesday":  first\_day=2  elif first\_day\_str=="wednesday":  first\_day=3  elif first\_day\_str=="thursday":  first\_day=4  elif first\_day\_str=="friday":  first\_day=5  elif first\_day\_str=="saturday":  first\_day=6  day\_num = (first\_day + date - 1) % 7  day = "Sunday"  if day\_num == 1:  day = "Monday"  elif day\_num == 2:  day = "Tuesday"  elif day\_num == 3:  day = "Wednesday"  elif day\_num == 4:  day = "Thursday"  elif day\_num == 5:  day = "Friday"  elif day\_num == 6:  day = "Saturday"  print(date, "day of the year with the first day", first\_day\_str, "is", day) | | | |
| **Output:**  First day of the year (in lowercase): tuesday  Which day number of the year do you want to know the day of (2 to 365): 19  19 day of the year with the first day tuesday is Saturday | | | |
| 7 | **Question:** Write a program that reads and integer N from the keyboard comptu8es and displays the sum of the numbers from N to 2\*N if N is non negative. If N is a negative number, then it’s the sum of the numbers from 2\*N to N. The starting and ending points are included in the sum. | | | |
| **Code:**  N = int(input("integer N: "))  sum=0  if N>=0:      for i in range(N, 2\*N+1):          sum+=i  else:      for i in range(2\*N, N+1, ):          sum+=i  print(sum) | | | |
| **Output:** | | | |
| integer N: -4  -30 | | integer N: 8  108 | |

# Chapter 2: Python Revision Tour-II

## Programs (pg 71)

|  |  |
| --- | --- |
| 2.4 | **Question:** Give three lists as list1 = [‘a’, ‘b’, ‘c’], list2 = [‘h’, ‘i’,‘t’] and list3 = [‘0’, ‘1’, ‘2’]. Write a program that adds individual elements of list2 and list3 to list1. The resultant list should be in the order of list3, list1 and then list2. |
| **Code:**  list1 = ['a', 'b', 'c']  list2 = ['h', 'i', 't']  list3 = ['0', '1', '2']  print("List 1:", list1)  print("List 2:", list2)  print("List 3:", list3)  for i in list3[::-1]:  list1.insert(0, i)  for i in list2:  list1.append(i)  print("Resultant List:", list1) |
| **Output:**  List 1: ['a', 'b', 'c']  List 2: ['h', 'i', 't']  List 3: ['0', '1', '2']  Resultant List: ['0', '1', '2', 'a', 'b', 'c', 'h', 'i', 't'] |
| 2.5 | **Question:** Write a program that finds an element’s index/position in a tuple without using index() |
| **Code:**  lst = eval(input("List: "))  element = eval(input("Element: "))  indices = []  for i in range(len(lst)):  if lst[i]==element:  indices.append(i)  print("Element's posible index/position: ", end="")  for i in indices:  print(i, end=" ") |
| **Output:**  List: [1,2,3,'a', 1]  Element: 1  Element's posible index/position: 0 4 |

## Type C

|  |  |
| --- | --- |
| 2 | **Question:** Write a program that should prompt the user to type some sentence(s) followed by “enter”. It should then print the original sentence(s) and the following statistics relating to the sentence(s): Number of words, Number of characters (including white-spaces and punctuation) and Percentage of characters that are alpha numeric. |
| **Code:**  sentences = input("Type a few sentence(s) (and press enter): ")  print("Original Sentence(s):")  print(sentences)  word\_count = len(sentences.split(" "))  char\_count = len(sentences)  alnum\_count = 0  for i in sentences:  if i.isalnum():  alnum\_count+=1  alnum\_percent = alnum\_count/char\_count \* 100  print("Number of words:", word\_count)  print("Number of characters:", char\_count)  print("Percentage of alpha-numeric characters: ",\ alnum\_percent, "%", sep="") |
| **Output:**  Type a few sentence(s) (and press enter): Large programs of size 100 MB to 200 MB are generally hard to store; except in case of integers. As, they have numeric values.  Original Sentence(s):  Large programs of size 100 MB to 200 MB are generally hard to store; except in case of integers. As, they have numeric values.  Number of words: 24  Number of characters: 126  Percentage of alpha-numeric characters: 78.57142857142857% |
| 3 | **Question:** Write a program that takes any two lists L and M of the same size and adds their elements together to form a new list N whose elements are sums of the corresponding elements in L and M. For instance, if L= [3, 1, 4] and M= [1, 5, 9], then N should equal [4, 6, 13]. |
| **Code:**  L = eval(input("List L: "))  M = eval(input("List M: "))  N = list()  for i in range(len(L)):  N.append(L[i]+M[i])  print("List N:", N) |
| **Output:**  List L: [3,1,4]  List M: [1,5,9]  List N: [4, 6, 13] |
| 8 | **Question:** Write a Python program that creates a tuple storing first 9 terms of Fibonacci series. |
| **Code:**  n=9  fib\_tup = (0,1)  for i in range(n-len(fib\_tup)):  fib\_tup+=sum(fib\_tup[-2:]),  print("Tuple with Fibonacci Series' (",n,"terms):", fib\_tup) |
| **Output:**  Tuple with Fibonacci Series' ( 9 terms): (0, 1, 1, 2, 3, 5, 8, 13, 21) |
| 9 | **Question:** Create a dictionary whose keys are month names and whose values are the number of days in the corresponding months.   1. Ask the user to enter a month name and use the dictionary to tell them how many days are in the month 2. Print out all the keys in alphabetical order 3. Print out all the months with 31 days 4. Print out the key-value pairs sorted by the number of days in each month |
| **Code:**  month\_days = {'January': 31, 'February': 28, 'March': 31, 'April': 30,  'May': 31, 'June': 30, 'July': 31, 'August': 31,  'September': 30, 'October': 31, 'November': 30, 'December': 31}  month\_name = input("Month Name:").strip().capitalize()  print("Number of days in", month\_name, "=", month\_days[month\_name])  keys = sorted(month\_days.keys())  print("All keys (months) in alphabetical order:")  print(", ".join(keys))  month\_31\_days = []  for k, v in month\_days.items():  if v==31:  month\_31\_days.append(k)  print("All months with 31 days:")  print(", ".join(month\_31\_days))  pairs = list(month\_days.items())  for i in range(1,len(pairs)):  if pairs[i][1]<pairs[i-1][1]:  for j in range(i):  if pairs[j][1]>pairs[i][1]:  pairs.insert(j, pairs[i])  pairs.pop(i+1)  print("All key-value (month-day) pairs sorted by number of days in each month:")  print(pairs) |
| **Output:**  Month Name:JANuary  Number of days in January = 31  All keys (months) in alphabetical order:  April, August, December, February, January, July, June, March, May, November, October, September  All months with 31 days:  January, March, May, July, August, October, December  All key-value (month-day) pairs sorted by number of days in each month:  [('February', 28), ('April', 30), ('June', 30), ('September', 30), ('November', 30), ('January', 31), ('March', 31), ('May', 31), ('July', 31), ('August', 31), ('October', 31), ('December', 31)] |

|  |  |
| --- | --- |
| 1 | **Question:** Question |
| **Code:**  Code |
| **Output:**  Output |