Sri Siddhartha Institute of Technology, Tumakuru
Department of Computer Science and Engineering

CS4LB3 : Python Programming Laboratory

Cycle-1 Experiments

1 a) Write a Python program to convert the given number of seconds to Hours, Minutes and Seconds and display the result.
(1 Hour = 3600 Seconds, 1 Minute = 60 Seconds)

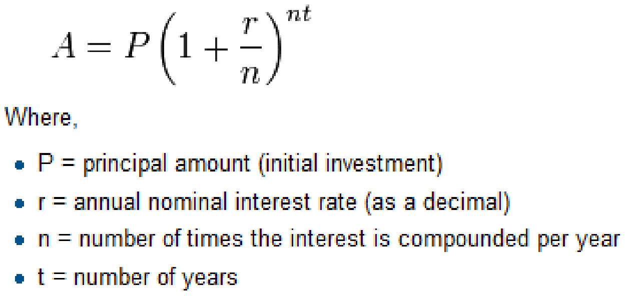
Program:

```
total_secs = int(input("Enter the total number of seconds:"))
hours = total_secs // 3600
secs_remaining = total_secs % 3600
minutes = secs_remaining // 60
secs_finally_remaining = secs_remaining % 60
print("Hrs=", hours, " Mins=", minutes,"Secs=", secs_finally_remaining)
```

Output:

```
Enter the total number of seconds: 4000
Hrs= 1  Mins= 6 Secs= 40
```

1 b) Write a Python program to compute and display the maturity amount for the deposit made in a bank by applying compound interest.



$$A = P\left(1 + \frac{r}{n}\right)^{nt}$$

Where,

- P = principal amount (initial investment)
- r = annual nominal interest rate (as a decimal)
- n = number of times the interest is compounded per year
- t = number of years

Program:

```
p=int(input("Enter the principal amount:"))
r=float(input("Enter the annual rate of interest(in decimals):"))
n=int(input("Enter the number of times the interest is compounded per year:"))
t=int(input("Enter the number of years of deposit:"))
a=p*(1+(r/n))**(n*t)
print("Maturity amount by applying compound interest is" , round(a,2))
```

Output:

```
Enter the principal amount: 100000
Enter the annual rate of interest (in decimals):0.06
Enter the number of times the interest is compounded per year:4
Enter the number of years of deposit:1
Maturity amount by applying compound interest is 106136.36
```

2 a) Write a Python program that uses Newton's method to find the square root of a given number.

Suppose that we want to know the square root of n. If we start with almost any approximation, we can compute a better approximation (closer to the actual answer) with the following formula:

better = (approximation + n/approximation)/2

Program:

```
n = float(input('Enter a number to find its square root using approximation method:'))
threshold = 0.001
approximation = n/2        # Start with some or other guess at the answer
while True:
   better = (approximation + n/approximation)/2
   if  abs(approximation - better) < threshold:
      print(round(better,2))
      break
   approximation=better
```

Output:

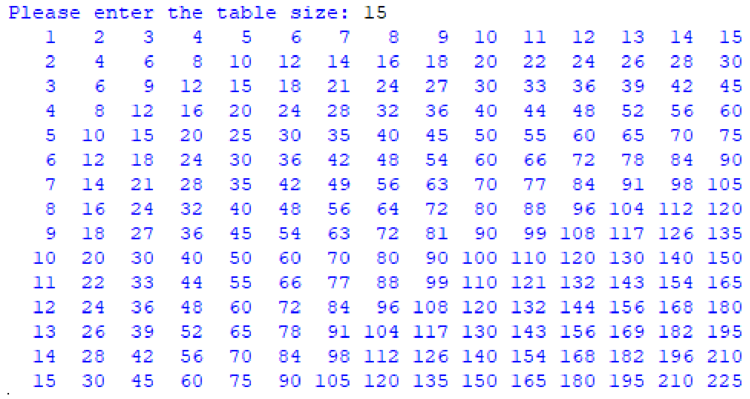
```
Enter a number to find its square root using approximation method:144
12.0
```

2 b) Write a Python program that generates multiplication table of given size (using nested for loops).

Program:

```
# Get the number of rows and columns in the table
size = int(input("Please enter the table size: "))

# Print a size x size multiplication table
for row in range(1, size + 1):
   for column in range(1, size + 1):
      product = row*column                          # Compute product
      print('{0:4}'.format(product), end='')        # Display product
   print()                                          # Move cursor to next row
```

Output:



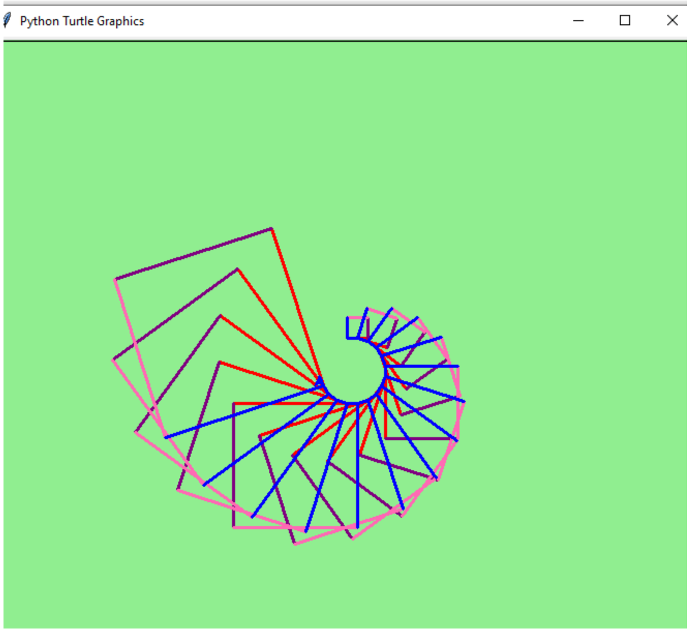
```
Please enter the table size: 15
   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
   2   4   6   8  10  12  14  16  18  20  22  24  26  28  30
   3   6   9  12  15  18  21  24  27  30  33  36  39  42  45
   4   8  12  16  20  24  28  32  36  40  44  48  52  56  60
   5  10  15  20  25  30  35  40  45  50  55  60  65  70  75
   6  12  18  24  30  36  42  48  54  60  66  72  78  84  90
   7  14  21  28  35  42  49  56  63  70  77  84  91  98 105
   8  16  24  32  40  48  56  64  72  80  88  96 104 112 120
   9  18  27  36  45  54  63  72  81  90  99 108 117 126 135
  10  20  30  40  50  60  70  80  90 100 110 120 130 140 150
  11  22  33  44  55  66  77  88  99 110 121 132 143 154 165
  12  24  36  48  60  72  84  96 108 120 132 144 156 168 180
  13  26  39  52  65  78  91 104 117 130 143 156 169 182 195
  14  28  42  56  70  84  98 112 126 140 154 168 182 196 210
  15  30  45  60  75  90 105 120 135 150 165 180 195 210 225
```

3 a) Write a Python program with a user defined function to draw multi coloured squares of given size using turtle graphics.

Program:

```
import turtle

def draw_multicolor_square(animal, size):
   """Make animal draw a multi-color square
   of given size."""
   for color in ["red", "purple", "hotpink", "blue"]:
      animal.color(color)
      animal.forward(size)
      animal.left(90)

window = turtle.Screen()  # Set up the window and its attributes
window.bgcolor("lightgreen")

tess = turtle.Turtle() # Create tess and set some attributes
tess.pensize(3)

size = 20 # Size of the smallest square
for _ in range(15):
   draw_multicolor_square(tess, size)
   size += 10 # Increase the size for next time
   tess.forward(10) # Move tess along a little
   tess.right(18) # and give her some turn

window.mainloop()
```

Output:



3 b) Write a Python program to compute the sum of the elements in a list using your own logic and also by calling the built-in sum function. Compute and display the time taken to find the sum in both the methods (Use time module).

Program:

```
import time

def do_my_sum(xs):
   sum = 0
   for v in xs:
      sum += v
   return sum

sz = 10000000  # Lets have 10 million elements in the list
testdata = range(sz)

t0 = time.perf_counter()
my_result = do_my_sum(testdata)
t1 = time.perf_counter()
print("my_result = {0} (time taken = {1:.4f} seconds)".format(my_result, t1-t0))

t2 = time.perf_counter()
their_result = sum(testdata)
t3 = time.perf_counter()
print("their_result = {0} (time taken = {1:.4f} seconds)".format(their_result, t3-t2))
```

Output:

```
my_result = 49999995000000 (time taken = 1.1111 seconds)
their_result = 49999995000000 (time taken = 0.5286 seconds)
```

4 a) Write a Python program to read a phrase, remove all punctuations in the phrase and display the phrase along with the list of words in the phrase.

Program:

```
import string
def remove_punctuation(phrase):
   phrase_without_punct = ""
   for letter in phrase:
      if letter not in string.punctuation:
         phrase_without_punct += letter
   return phrase_without_punct

my_story = """India is my country, all Indians are my "brothers and sisters"; I love my country!"""
result=remove_punctuation(my_story)
print("Phrase without punctuations:\n" + result)
words = result.split()
print("\n List of words in the phrase:\n", words)
```

Output:

```
Phrase without punctuations:
India is my country all Indians are my brothers and sisters I love my country

List of words in the phrase:
 ['India', 'is', 'my', 'country', 'all', 'Indians', 'are', 'my', 'brothers', 'and', 'sisters', 'I', 'love', 'my',
'country']
```

4b) Write a Python program to sort a list of tuples based on the sum of elements in the tuple (Use lambda function to generate the key for sorting)

Examples:

Input: [(4, 5), (2, 3), (6, 7), (2, 8)]
Output: [(2, 3), (4, 5), (2, 8), (6, 7)]

Input: [(3, 4), (7, 8), (6, 5)]
Output: [(3, 4), (6, 5), (7, 8)]

Program:

```
# Input list initialisation
list_of_tuples = [(4, 5), (2, 3), (6, 7), (2, 8)]

print("The original list of tuples is ")
print(list_of_tuples)

# Passing lambda as key to sort list of tuple
list_of_tuples.sort(key = lambda x: x[0] + x[1])

# Printing output
print("\n Sorted list of tuples based on sum\n", list_of_tuples)
```

Output:

```
The original list of tuples is
[(4, 5), (2, 3), (6, 7), (2, 8)]

Sorted list of tuples based on sum
 [(2, 3), (4, 5), (2, 8), (6, 7)]
```