|  |
| --- |
| **Program 01** |
| **Output** |
| 1  6  0  15 |
| **Source Code** |
| #This program takes any number of arguments and then sums them  #and then returns the sum printed  def main():  add(1)  add(1,2,3)  add()  add(1,2,3,4,5)  #This function sums any given number of arguments  def add(\*args):  total = 0  for n in args:  total += n  print(total)    if \_\_name\_\_=="\_\_main\_\_": main() |

|  |
| --- |
| **Program 02** |
| **Output** |
| 10  7  5  3  2 |
| **Source Code** |
| #This program takes 0-4 arguments and then sums  #the numbers passed and the returns that value  #and the sum is then printed in main  def main():  print(add(1,2,3,4))  print(add(1,2,3))  print(add(1,2))  print(add(1))  print(add())  #Add method that takes 0-4 arguments and uses a series  #of If-Elif-Else statements to correctly sum up the  #arguments not including the None value.  def add(first = None, second = None, third = 1, fourth = 1):  sum = 0  if first is None and second is None:  sum = third + fourth  return sum  elif first is None:  sum = second + third + fourth  return sum  elif second is None:  sum = first + third + fourth  return sum  else:  sum = first + second + third + fourth  return sum  if \_\_name\_\_=="\_\_main\_\_": main() |

|  |
| --- |
| **Program 03** |
| **Output** |
| The highest score was: ann 100  The highest score was: ann 100  The highest score was: ann 100 |
| **Source Code** |
| #This program utilizes a dictionary of names and associated  #values and utilizes key-word arguments to go through all  #of the items and search for the highest score and return  #that score and the student associated with that score  def main():  highScore(ann=100,bob=90,carol=80,dave=70)  highScore(ann=100,bob=90,carol=80,dave=70, ed = 69)  highScore(ann=100,bob=90)  #Function for finding the highest score and the student with that score  def highScore(\*\*kwargs):  maximum = 0  for i in kwargs:  if kwargs[i] > maximum:  maximum = kwargs[i]  index = i  print('The highest score was: {} {}\n'.format(index, maximum))    if \_\_name\_\_=="\_\_main\_\_": main() |

|  |
| --- |
| **Program 04** |
| **Output** |
| 30  31  30  31  28  31  30  31  30  31  31  30  31  30  31 |
| **Source Code** |
| #This program utilizes a function called Month Days that  #takes 0-2 arguments. The first argument is the desired  #month to start at 0 <-> January, 1 <-> February and so on.  #The next argument is the number of months you wish to cycle  #through. The program then prints the number of days in all  #the associated months requessted.  def main():  for i in monthDays(3,3):  print(i)  print()  for i in monthDays():  print(i)  print()  #Month Days Method  def monthDays(start = 0, months = 12):  if (start+months) <= 12:  daysList = [31,28,31,30,31,30,31,31,30,31,30,31]  for i in range(start, start+months):  yield daysList[i]  else:  #Raised if start and months sum to greater than 12.  raise IndexError('The sum of the arguments cannot exceed twelve(12).')    if \_\_name\_\_ == "\_\_main\_\_": main() |

|  |
| --- |
| **Program 05** |
| **Output** |
|  |
| **Source Code** |
|  |