**Practical session 8: Continue work on the functions**

*For the practical parts of this lab (implementing programs, running them) please save the Python programs that you create and take screenshots of the execution (evaluation) of your programs. Commit (upload) all source code you create to your code repository.*

**Tasks:**

|  |
| --- |
| Consider this function:  **def mystery(x, y) :**  **result = (x + y) / (y - x)**  **return result**  What is the result of the call mystery(2, 3)? |
| What does this program print?  **def main() :**  **a = 5**  **b = 7**  **print(mystery(a, b))**  **def mystery(x, y) :**  **z = x + y**  **z = z / 2.0**  **return z**  **main()** |
| What does this program print?  **def main() :**  **a = 4**  **print(mystery(a + 1))**  **def mystery(x) :**  **y = x \* x**  **return y**  **main()** |
| Consider this function that prints a page number on the left or right side of a page:  **if page % 2 == 0:**  **print(page)**  **else :**  **print("%60s%d" % (" ", page))**  Introduce a function that returns a Boolean to make the condition in the if statement easier to understand. |
| Transform the following instructions into a function called **count\_spaces**. Define a **main** function that will ask the user to enter some input and call the **count\_spaces** function to return the number of spaces.  **# Counts the number of spaces**  **spaces = 0**  **for char in userInput :**  **if char == " " :**  **spaces = spaces + 1** |
| Consider this recursive function:  **def mystery(n) :**  **if n <= 0 :**  **return 0**  **else:**  **return n + mystery(n - 1)**  What is mystery(4)? |
| **def mystery(n) :**  **if n <= 0 :**  **return 0**  **else:**  **return mystery(n // 2) + 1**  What is mystery(20)? |
| Consider these functions:  **def f(x) :**  **return g(x) + math.sqrt(h(x))**  **def g(x) :**  **return 4 \* h(x)**  **def h(x) :**  **return x \* x + k(x) - 1**  **def k(x) :**  **return 2 \* (x + 1)**  Without actually compiling and running a program, determine the results of the following function calls:  a. x1 = f(2)  b. x2 = g(h(2))  c. x3 = k(g(2) + h(2))  d. x4 = f(0) + f(1) + f(2)  e. x5 = f(-1) + g(-1) + h(-1) + k(-1) |
| Consider the following function:  **def f(a) :**  **if a < 0 :**  **return -1**  **n = a**  **while n > 0 :**  **if n % 2 == 0 : # n is even**  **n = n // 2**  **elif n == 1 :**  **return 1**  **else :**  **n = 3 \* n + 1**  **return 0**  Perform traces of the computations f(-1), f(0), f(1), f(2), f(10), and f(100). |