**Lab 9 - Functions**

**Content:** Python functions, if-else statements, loops, strings

**Questions:**

**1.** Write a function that takes a number as a parameter and prints the

numbers from 1 to that number on the screen.

**2.** Write a function that takes a number as a parameter and iterates from 0

to that number. For each iteration, it will check if the current number is even or odd,

and report that to the screen (e.g. “1 is odd”, “2 is even”).

**3.** Write a function that takes a number as a parameter, iterates from 0 to

that number, and for each iteration of the loop, multiplies the current number by 9

and prints the result (e.g. “2 \* 9 = 18”).

**4.** Write a function that asks the user for a number and prints the sum of all

numbers from 1 to the number they enter.

**5.** Write a function to print a factorial of a number.

**6.** Write a function that takes a string as a parameter and returns a string that is made up of the first two characters and the last two characters. If the string has a length less than 4 the program prints a message on the screen. For example: “hello there” will result in “here”

**7.** Write a Python program to remove the characters which have odd index

values of a given string. The function should return the new string.

**8.** Write a Python function to get the first half of a specified string of even length.

Sample function and result:

string\_first\_half(“Python”)

should return Pyt

**9.** Write a Python function to insert a string in the middle of a string.

Sample function and result:

insert\_string\_middle(“{{}}”,”PHP”)

Should return {{PHP}}

**10.** Write a Python function that takes a string and two indices, and returns

a string with the part between the indices removed.

For example: remove\_substring(“Hello there”, 2, 6) should return “Hehere”

**11.** Heap of Beans:

We are going to play a game called the heap of beans. You start with a heap of beans (we will start with 16 beans) and two players. Each player can remove 1, 2, or 3 beans from the pile of 16. Each player is required to take some number of beans from the pile during each turn. The players take turns removing beans, and the player who takes the last bean from the pile is the loser. You can try this game with a partner using 16 pieces of paper as beans.

Each player is to be represented by a function, so we will have two functions. Each function takes a single argument, representing the present number of beans, and returns the number of beans remaining after the player function takes its turn. During the operation of the player function, the function reports (prints) how many beans were removed. Note that the function decides how many beans to remove. It takes no input from the user.

You also need a main program. The main program initializes the heap of beans to

16 and then alternately calls the first player function and the second player function

until one of them gets the last bean. The main then reports who the loser is and exits.

(a) Write a simple player function such as one that always takes exactly one bean. It

isn’t very interesting, but it lets you test your program.

(b) Now for the fun part. Write a “smart player” function and test it against other

students’ functions in your class.