The University Of Sheffield.

The University of Sheffield CiCS Research IT

PRACTISE SESSION 1 SOLUTIONS

1. Try different conversions, such as string to integer or float. Integer to string, etc… Some of them will work some of them will not. Why?

```
# Define the following variables for test

number_1 = 100
number_2 = 3.14
name = ‘John’
```

1. Try different conversions, such as string to integer or float. Integer to string, etc… Some of them will work some of them will not. Why?

```
# INT TO …

test = float(number_1) # OK: INT -> FLOAT
test = int(number_1)   # OK: INT -> INT
test = str(number_1)   # OK: INT -> STR

# FLAOT TO …

test = float(number_2) # OK: FLOAT -> FLOAT
test = int(number_2)   # OK: FLOAT -> INT
test = str(number_2)   # OK: FLOAT -> STR
```

1. Try different conversions, such as string to integer or float. Integer to string, etc… Some of them will work some of them will not. Why?

```
# STRING TO …

test = float(name) # Cond: STR -> FLOAT
test = int(name)   # Cond: STR -> FLOAT
test = str(name)   # OK: STR -> STR
```

2. Try the basic arithmetics and calculate the average and the standard deviation of the following numbers: 3, 7, 32, 54, 12, 34. You can create 6 variables now.

```
#Define the variables first.

a = 3
b = 7
c = 32
d = 54
e = 12
f = 34

# Calculate the mean.
avg = (a + b + c + d + e + f) / 6

# Print it.
print(avg)
```

```
# Calculate the standard deviation.
st_a = (a - avg) ** 2
st_b = (b - avg) ** 2
st_c = (c - avg) ** 2
st_d = (d - avg) ** 2
st_e = (e - avg) ** 2
st_f = (f - avg) ** 2

std = st_a + st_b + st_c + st_d + st_e + st_f
std = (std / 6) ** (1/2)

print(std)
```

3. Write a program which is able to convert distance in miles to kilometers. The structure of your output will be,

Converting distance in miles to kilometers:

Distance in miles:

<?>

Distance in kilometers:

<?>

```
# Read the input for keyboard.
miles = input('Please, type a number: ')

# Type conversion
miles = float(miles)

# Calculate the result
km = miles * 1.6

# Print it.
print('Distance in miles:')
print(miles)
print('Distance in kilometers:')
print(km)
```

```
# Read the input for keyboard.
km = input('Please, type a number: ')

# Type conversion
km = float(km)

# Calculate the result
miles = km / 1.6

# Print it.
print('Distance in kilometers:' + str(km))
print('Distance in: miles' + str(miles))
```

The University Of Sheffield.

The University of Sheffield CiCS Research IT

PRACTISE SESSION 2 SOLUTIONS

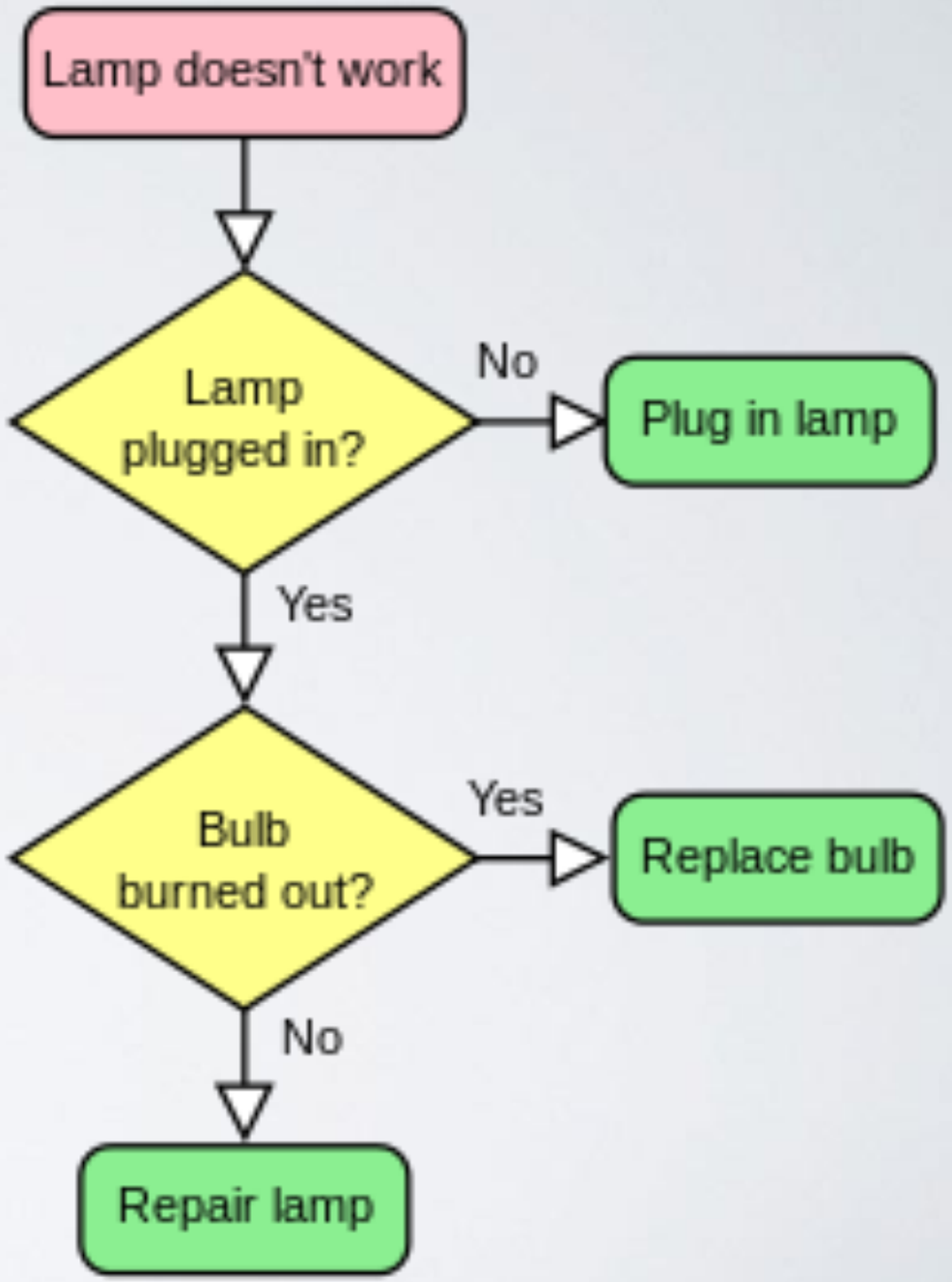
1) Write a program that asks the user how old they are and tells them if they are old enough to vote. Then extend the program to tell them how many years they have until retirement (assume to be 65).

```
# Read the input for keyboard.
age = input('How old are you: ')

age = int(age)

if age >= 18:
  print('You are allowed to vote.')

print('You have ' + str(65-age) +' year(s) until retire.')
```

The University Of Sheffield.

The University of Sheffield CiCS Research IT

2) The following flowchart shows the underlying logic of a simple system you will implement. The program asks two questions: Lamp plugged in and Bulb burned out? The answer should be a number (0-no, 1-yes). The green boxes indicate the potential outputs.



```
answer1 = input('Lamp plugged in? (0-no, 1-yes): ')
answer1 = int(answer1)

if answer1 == 0:
  print('Plug the lamp.')
else:
  answer2 = input('Bulb burned out? (0-no, 1-yes): ')
  answer2 = int(answer2)
  if answer2 == 0:
    print('Repair lamp.')
  else:
    print('Replace bulb.')
```

3) Print a sequence of numbers between 4 and 20. Print all numbers first, then display the even numbers.

```
for i in range(4,20):
  print(i)

for i in range(4,20,2):
  print(i)
```

4) Define a sequence of numbers, $x_n=n^2+1$, for integers n=0,1,2,…,N. Write a program that displays x_n for n=0,1,…,N using a for loop. (n^2 = n**2). N should be defined by the user.

```
# Input from keyboard
n = input('Number? ')

# Type conversion
n = int(n)

# For loop until i reaches n
for i in range(n):
  number = (i ** 2) + 1
  print(number)
```

Write a program that takes three numbers from the user and displays the largest of them.

```
# Input from keyboard
a = input('Number one? ')
b = input('Number two? ')
c = input('Number three? ')

if a > b and a > c:
  print(a)

if b > b and b > c:
  print(b)

if c > a and c > b:
  print(c)
```

The University Of Sheffield.

The University of Sheffield CiCS Research IT

SESSION 2

The University Of Sheffield.

The University of Sheffield CiCS Research IT

FUNCTIONS

- A function is a block of organised and reusable code.
- Functions provide better modularity for your application and a high degree of code reusing.


```
def functionname(parameters):
    #Your code in this function.
    return expression
```

The following function takes a number as an input parameter and returns the area of a circle.

```
def circle_area(radius):
    Area = 3.14 * (radius ** 2)
    return Area
```

CALLING A FUNCTION

Once the basic structure of a function is finalised, you can execute the function by calling it from another function or directly from the Python prompt.

```
result = circle_area(5)
```

```
# Define your function
def circle_area(radius):
    Area = 3.14 * (radius ** 2)
    return Area

# Now you can call circle_area.
result = circle_area(5)

# Print the result
print(result)
```

The University Of Sheffield.

The University of Sheffield CiCS Research IT

Study Drill

Write a function named is_positive(n). The function tests whether the number is positive and returns the value True otherwise returns the value False. The parameter n is defined by the user.

```
# Define your function first
def is_positive(n):
    if n >= 0:
        return True
    else:
        return False

# Read the input number from keyboard
n = input('Type a number: ')

# Now you can call your function.
result = is_positive(n)

# Print the result
print(result)
```

MULTIPLE PARAMETERS

```
def foo(par1, par2, par3):
    # Do something here and return it
    return Area
```

```
result = foo(5, 4.13, “hi”)
```

Can have as many parameters as you need.

MULTIPLE RETURNS

```
def multiple(par):
    a = 3.14 * par     # Float and int. OK
    b = 100 * par      # Int and int. OK
    c = “Hi.” * par    # Str and int. OK?
    return a, b, c
```

```
res1, res2, res3 = multiple(2)
```

Study Drill

This is the input:

```
>>> Please type the first number:  3
>>> Please type the next number:  5
```

and the desired output is,

```
3.0 + 5.0 = 8.0
3.0 - 5.0 = -2.0
3.0 * 5.0 = 15.0
3.0 / 5.0 = 0.6
```

Reproduce it.

```
def calculator(num1, num2):
    # This function do the basic arithmetics.
    plus = num1 + num2
    minus = num1 - num2
    multi = num1 * num2
    div = num1 / num2
    return plus, minus, multi, div

# Input
num1 = float(input('Please type the first number: '))
num2 = float(input('Please type the next number: '))

# Call your function
plus, minus, multi, div = calculator(num1, num2)

# Print the output
print(num1, '+', num2, '=', plus)
print(num1, '-', num2, '=', minus)
print(num1, '*', num2, '=', multi)
print(num1, '/', num2, '=', div)
```

GLOBAL VS LOCAL VARIABLES

If a variable is defined outside of any function, it is assumed to be a **global variable**.

If a variable is defined anywhere within a function, it is assumed to be a **local variable**.

GLOBAL VS LOCAL VARIABLES

```
# This prints a line. No return here.
def test_function():
    print(s)

# Define a global variable
s = "I hate spam”

# Call the test_function.
test_function()
```

GLOBAL VS LOCAL VARIABLES

The code works because variable 's' was assigned outside of the function.

```
s = "I hate spam"
```

Hence, variable 's' is global and visible everywhere.

GLOBAL VS LOCAL VARIABLES

Now try something else.

```
def test_function():
    s = "I hate spam”

test_function()
print(s)
```

Does it work?

GLOBAL VS LOCAL VARIABLES

You should have got the following error:

```
Traceback (most recent call last):
  File "python", line 5, in <module>
NameError: name 's' is not defined
```

The variable ’s’ was assigned inside of a function. Hence, it was local. It will not be visible outside of the function.

GLOBAL VS LOCAL VARIABLES

If you defined a variable within a function, return the variable if you need it later.

```
def test_function():
    s = "I hate spam”
    return s

result = test_function()
print(result)
```

The University Of Sheffield.

The University of Sheffield CiCS Research IT

PRACTICE SESSION 3

LIST

The University Of Sheffield.

The University of Sheffield CiCS Research IT

The list is a versatile datatype available in Python which can be written as a list of comma-separated values (items) between square brackets.

```
list1 = ['physics', 'chemistry', 1997, 2000];
list2 = [1, 2, 3, 4, 5 ];
list3 = ["a", "b", "c", "d"]

print(list1)
print(list2)
print(list3)
```

ACCESSING VALUES IN LISTS

Use the square brackets for slicing along with the index.

```
list1 = ['physics', 'chemistry', 1997, 2000];

print(list1[1])
print(list1[0])
print(list1[3])
print(list1[2])
```

UPDATING LISTS

```
# Define a list first.
list = ['physics', 'chemistry', 1997, 2000]

# Print it.
print(list)

# Modify only one element of the list.
list[2] = 2001

# Print it again.
print(list)
```

DELETE AND APPEND

```
# Define a list first.
list = ['physics', 'chemistry', 1997, 2000]

# Print it.
print(list)

# Delete an element.
del list[2]

# Print it.
print(list)

# Append it
list.append(2005)

# Print it again.
print(list)
```

BASIC OPERATION

Python Expression	Results	Description
len([1, 2, 3])	3	Length
[1, 2, 3] + [4, 5, 6]	[1, 2, 3, 4, 5, 6]	Concatenation
['Hi!'] * 4	['Hi!', 'Hi!', 'Hi!', 'Hi!']	Repetition
3 in [1, 2, 3]	TRUE	Membership

The University Of Sheffield.

The University of Sheffield CiCS Research IT

ITERATION

```
# Create a new list
list = [1, 2, 3]

# Iteration 1
for element in list:
    print(element)

# Iteration 2
for iter in range(len(list)):
    print(list[iter])
```

The University Of Sheffield.

The University of Sheffield CiCS Research IT

1. Write a function for calculating the average of a list. Your input is [1,2,3,4,5,6,7,8,9]. The output should be 5.

2. Write a function to check whether a list is empty or not. Define two lists: [1,2,3,4,5,6,7,8,9] and []. If the list is empty, print the string 'empty list'. If it is not then print the string 'list is not empty'.

The University Of Sheffield.

SOLUTION 1

The University of Sheffield CiCS Research IT

```
def sum_list(items):
    sum_numbers = 0
    for x in items:
        sum_numbers = sum_numbers + x
    return sum_numbers / len(items)

# Define the input list
list = [1,2,3,4,5,6,7,8,9,10]
result = sum_list(list)
print(result)
```

SOLUTION 2

```
def empty_list(list_to_check):
  if len(list_to_check) == 0:
    return 'Empty list.'
  else:
    return 'The list is not empty.'

# Define the input list
list_test_1 = [1,2,3,4,5,6,7,8,9]
list_test_2 = []

# Check the first list
result = empty_list(list_test_1)
print(result)

# Check the next list
result = empty_list(list_test_2)
print(result)
```

The University Of Sheffield.

The University of Sheffield CiCS Research IT

PRACTICE SESSION 4