Exercise 01

Saturday, October 21, 2023 9:50 PM

Problem 1 to hand in: *Hand in*

The following pseudocode and Python code contain errors. For each subtask determine the error (i.e., explain what the error is) and write down how it can be repaired.

a) Given two positive integers a and b, the code should return the remainder c of the integer division $\lfloor \frac{a}{b} \rfloor$, i.e. $c = a - b \cdot \lfloor \frac{a}{b} \rfloor$.

```
get_remainder(a, b):
1 while a > b do
2 |   a ← a − b
3 return a
```

b) Given an integer n, the following pseudocode should return a Boolean value `True` if and only if n is divisible by 3.

```
is_even(n):
1 if n ≡ 0 mod 3 then
2 |   result ← True
3 result ← False
4 return result
```

c) Given two integers m and n, the following Python code should compute the power m^n.

```
1    def power(m, n):
2        power = 1
3        for index in range(n):
4            power = power * m
5        return power
```

ranging 0 ~ n-1

n steps in total

d) Given a positive integer n, the following Python code should compute $\sum_{i=1}^{n} i^2$.

```
6    def sum_squares(n):
7        sum = 0
8        for index in range(n):
9            sum = sum + index**2
10       return sum
```

+1

a) The current pseudocode will return an $a < 0$, which is clearly not the remainder.

The deduction shall stop as soon as

The deduction shall stop as soon as

$a < b$ ($a // b = 0$)

b) In the code. either the condition is fullfilled or not, the statement at line 3 :

result ← False

will always be executed. This means that the function will always return false, regardless of the input. The declaration of the variable result shall be positioned before the condition.

Additionally, I believe the function shall not be called is_even and $n \bmod 3 \equiv 0$ might be a typo

c) According to the code and the writing style. The statement return lies within the for loop. The algorithm returns power (=m) in the very first step. Putting the return statement outside the loop will give the correct value.

d) This refers to a typical Python mistake. The command range(n) creates an array from 0 to n-1. To get a correct output, one must use range(n+1)

Since the first index is 0, it does not affect the total sum when the permutation starts from 0.

the total sum when the permutation starts from 0.
We may leave it as it is.

Problem 2 for discussion: *Efficiency and Refactoring*

Take a closer look at the code examples in Exercise 1.

a) What are their running times?

b) How can they be improved in terms of readability and brevity?

a) The running times vary with computer hardware setup.

- a) $O(n)$, $n \sim a // b$
- b) $O(1)$
- c) $O(n)$
- d) $O(n)$

b) . a

```python
# this is Python code
a % b
```

- b

```python
# this is Python code
def divisible_by_3 (n):
    return n % 3 == 0
```

- c

```python
m ** n
```

- d index to i for readability