STRINGS
Retake of test

- Format is the same as in the main test
- Best attempt counts
- Possible date and time: Friday, October 18, 14:15-18:00.
Project

• Register your team
• Submit project description
• Schedule
  • Form a team and select topic: October 14
  • Alpha version: November 11
  • Beta version: December 9
Using the functions add and multiply, rewrite the expression

\[ S = a + b \times (c + d) \]

in such a way that the result doesn't contain any multiplication signs and any addition signs.
The variable $x$ has an integer value. Write a piece of Python code that finds out whether $x$ is odd or even, and in the latter case, additionally, finds whether $x$ is divisible by 4 or not.

```python
_______:
    print("x is odd")
else:
    print("x is even, and more precisely:")
_______:
    print("x is even and divisible by 4")
_______:
    print("x even and not divisible by 4")
```
Test review

- Car worth
- Jubilees
String data type

```python
>>> type("mg5#X4")
<class 'str'>
>>> type('mg5#X4')
<class 'str'>
```

Single or double quotes
Conversion to numeric type and back

\[
\begin{align*}
\text{int}'12'\text{) \quad \rightarrow \quad & 12 \\
\text{float}'34.5'\text{) \quad \rightarrow \quad & 34.5 \\
\text{str}(12) \quad \rightarrow \quad & '12' \\
\text{str}(34.5) \quad \rightarrow \quad & '34.5' \\
\end{align*}
\]
Indices

characters  →  C e d a r
indices     →  0 1 2 3 4

word = "Cedar"
letter = word[1]  →  e
print(letter)

print(word[5])  →  Traceback (most recent call last):
print(word[5])
IndexError: string index out of range
Length

characters → Cedar
indices → 0 1 2 3 4

word = "Cedar"
length = len(word) → 5
print(length)
Looping through strings

• Loop over indices

```python
word = 'Flick'
index = 0
while index < len(word):
    letter = word[index]
    print(index, letter)
    index = index + 1
```

• Loop over characters

```python
word = 'Flick'
for letter in word:
    print(letter)
```
Slicing

characters    indices

>>> text = 'Clear cache'

>>> text[3:8]
'ar ca'

>>> text[3:-2]
'r cache'

>>> text[::9]
'Clear cac'

>>> text[:]
'Clear cache'

>>> text[::]
Operations with strings

- **Concatenation +**
  
  ```python
  >>> space = " "
  >>> "Talk" + space + "show"
  'Talk show'
  ```

- **Multiplication ***
  
  ```python
  >>> ("Talk" + space) * 3
  'Talk Talk Talk '
  ```
Logical expressions using strings

• Substring

```python
>>> "lo" in "Hello"
True
>>> "a" in "Hello"
False
```

• Equality

```python
>>> "hello" != "Hello"
True
>>> "hello" == "Hello"
False
```

• Inequality

```python
>>> "world" > "Hello"
True
>>> "world" < "Hello"
False
```
## Functions

- capitalize
- casefold
- center
- count
- encode
- endswith
- expandtabs
- find
- format
- format_map
- index
- isalnum
- isalpha
- isdecimal
- isdigit
- isidentifier
- islower
- isnumeric
- isprintable
- isspace
- istitle
- isupper
- join
- ljust
- lower
- lstrip
- maketrans
- partition
- replace
- rfind
- rindex
- rjust
- rpartition
- rsplit
- rstrip
- split
- splitlines
- startswith
- strip
- swapcase
- title
- translate
- upper
- zfill

[https://docs.python.org/3/library/stdtypes.html#string-methods](https://docs.python.org/3/library/stdtypes.html#string-methods)