Introduction to programming

Labor04

Revision

- `match` statement
- Loops
  - `range()` function
  - `for` loop
  - `while` loop

Homework 1

- **Maximum of Integer Numbers #1**
- https://progcont.hu/progcont/b50/?pid=200309

Solution – 1

```
n = int(input())
num = int(input())
max = num

for i in range(1, n):
    num = int(input())
    if num > max:
        max = num

print(max)
```

Solution – 2

```
n = int(input())
first = True

for i in range(n):
    num = int(input())
    if first == True:
        max = num
        first = False
    if num > max:
        max = num

print(max)
```

Homework 2

- **Grading #3**
- https://progcont.hu/progcont/b50/?pid=200316

Grading #3

```
point = int(input())

while point >= 0:
    if point >= 80:
        print("jeles")
    elif point >= 70:
        print("jo")
    elif point >= 60:
        print("kozepes")
    elif point >= 50:
        print("elegseges")
    else:
        print("elegtelen")
    point = int(input())
```

Grading #3

```
while True:
    point = int(input())
    if point < 0:
        break
    if point >= 80:
        print("jeles")
    elif point >= 70:
        print("jo")
    elif point >= 60:
        print("kozepes")
    elif point >= 50:
        print("elegseges")
    else:
        print("elegtelen")
```

Homework 3

- **Average of Integers #1**
  https://progcont.hu/progcont/b50/?pid=200304

- **Average of Integers #3**
  https://progcont.hu/progcont/b50/?pid=200306

Average of Integers #1

```
sum = 0
n = int(input())

for i in range(n):
    num = int(input())
    sum += num

print("{0:.2f}".format(sum / n))
```

Average of Integers #3

```
sum = 0
c = 0
n = int(input())

while n != 0:
    sum += n
    c += 1
    n = int(input())

print("{0:.2f}".format(sum / c))
```

The break statement

- with the break statement we can stop the loop even if the while condition is true
- the while, for loop is terminated

- Example:

```
i = 0
while True:   # infinite loop
    i += 1
    if i == 10:
        break #end of the loop
    print(i)
```

The continue statement

- interrupts the current loop iteration and continues by evaluating the loop condition.
- the while, for loop conditions are evaluated
- the part after `continue` is completely omitted in the current run

Exercise

- Write a program that reads 10 integer numbers and calculates the sum of the evens.

Solution – 1

```
i = 0
s = 0
while i < 10:
    n = int(input("n = "))
    i += 1
    if n % 2 != 0:
        continue
    s = s + n

print("Sum of evens= ", s)
```

Solution – 2

```
i = 0
s = 0
while True:
    if i == 10:
        break
    i += 1
    n = int(input("n = "))
    if n % 2 != 0:
        continue
    s = s + n
```

Input numbers/Read values

- Until 0
- Until a negative number
- Until a positive number
- Until an even number
- Until a given condition
- Until EOF (CTRL+D)
- etc.

Input numbers until 0

```
n = int(input("n = "))

while n != 0:                    # while n:
    n = int(input("n = "))
```

Input numbers until 0

```
while True:
    n = int(input("n = "))
    if n == 0:
        break
```

Input numbers until 0

```
while 1:
    n = int(input("n = "))
    if n == 0:
        break
```

Input numbers until a negative number

```
n = int(input("n = "))
while n >= 0:
    n = int(input("n = "))
```

Input numbers until a negative number

```
while True:
    n = int(input("n = "))
    if n < 0:
        break
```

Input numbers until an even number

```
while True:
    n = int(input("n = "))
    if n % 2 == 0:
        break
```

Exercise

- Input integer numbers continuously until 0. Count and print out how many negative and positive numbers are input.

Solution

```
pos = 0
neg = 0
while True:
    n = int(input("n="))
    if n == 0:
        break
    if n > 0:
        pos += 1
    if n < 0:
        neg += 1
print("Positive numbers = ", pos)
print("Negative numbers = ", neg)
```

Exercise

- Write a program that inputs integer numbers from the keyboard until the sum of the numbers exceeds 100.
- At the end of the input, print out how many of the numbers were odd and how many were even.

Solution

```
s = 0
even = 0
odd = 0
while True:
    n = int(input("n="))
    s += n
    if s > 100:
        break
    if n % 2 == 0:
        even += 1
    else:
        odd += 1
print("Number of evens = ", even)
print("Number of odds = ", odd)
```

Exceptions

- It can be used when an unexpected error occurs::
  - try-except
  - try-finally
- **Exceptions** are operations that the interpreter or compiler performs when it detects an error during program execution.
- As a general rule, program execution is **terminated** and a more or less explicit error message is displayed.

Exceptions

- print(5/0)
  - ZeroDivisionError: division by zero

- Python's exception handling mechanism uses the try – except – else statement set, which allows you to catch an error and execute a snippet of code specific to that error.

- This works as follows :
  - The block of statements following try is executed conditionally by Python. If an error occurs when one of the statements is executed, Python deletes the errored statement and executes the block of code following the except instead.
  - If no error occurs in the statements following try, it executes the code block following else (if that statement is present). In either case, execution of the program may continue with the subsequent.

Try – Except

```
try:
    ...
except (NameError,TypeError):
    ...
except:
    print('An unexpected exception has occurred')
else:
    print('No exception occurred')
```

Types of exceptions

- `ValueError`: for conversions (e.g. int("apple"))
- `TypeError`: for type errors (e.g. "apple" /4)
- `ZeroDivisionError`: division by zero (e.g. 5/0)
- `IndexError`: over-indexing
- `EOFError`: end of file (EOF)
- `ModuleNotFoundError`: import on non-existent module
- `NameError`: for non-existent variable

Try – Finally

- It often happens that, whether an exception is thrown or not, something has to be done by the program, such as closing a window, disconnecting from the Internet, closing a file, and then it can be used.

```
try:
    x = int(input("Input an integer number: "))
    print(x, "Last digit:", x % 10)
except ValueError as e:
    print("Error:", e)
except EOFError:                # no data (Ctrl+D)
    print("End-of-file sign.")
finally:
    print("This is always.")
```

Reading data until EOF

- **EOF** – End Of File
- PyCharm/Linux: CTRL+D
- Windows: CTRL+Z

```
import sys
for line in sys.stdin:
    print(line)

while True:
    try:
        line = input()
        print(line)
    except EOFError:
        break
```

Average of integers #2

```
import sys

sum = 0
c = 0

for n in sys.stdin: #EOF - CTRL+D
    sum += int(n)
    c += 1
print("{0:.2f}".format(sum / c))
```

Average of integers #2

```
sum = 0
c = 0
while True:
    try:
        sum += int(input())
        c += 1
    except EOFError: #EOF - CTRL+D
        break
print("{0:.2f}".format(sum / c))
```

Strings

```
> s = 'Hello'
> len(s) -> 5
> s[4] -> 'o'
> s[0] -> 'H'
> s+'!’ -> 'Hello!'
> s.lower() ->'hello'
> s.upper() -> 'HELLO'
> s.find('e') -> 1
> s.find('a') -> -1
> s='    jhasgjhsf    '
> s.strip() -> 'jhasgjhsf'
```

Strings

```
> s='1234'
> s.isdigit() -> True
> s.startswith('12') -> True
> s.endswith('12') -> False
> s.replace('12','34') -> '3434'
> s="apple pear orange"
> s.split() -> ['apple', 'pear', 'orange']
> s="apple:pear:orange"
> s.split(":") -> ['apple', 'pear', 'orange']
> s.split("a") -> ['', 'pple:pe', 'r:or', 'nge']
> "--".join(["apple", "pear", "orange"]) ->
  'apple--pear--orange'
```

String fuctions

- **s.lower(), s.upper()** – return lowercase, uppercase version of string
- **s.strip()** – strips whitespace characters from the beginning and end of the string
- **s.isalpha() / s.isdigit() / s.isspace()...** – checks if all characters of the string belong to the given character class
- **s.startswith('other'), s.endswith('other')** – checks if the string starts / ends with the other string
- **s.find('other')** – Returns whether the string contains the other string (not specified as a regular expression). If so, returns the index of the first character of the first occurrence. If no, the return value is -1.

String fuctions

- **s.replace('old', 'new')** – replaces all occurrences of 'old' in the string with 'new'
- **s.split('delim')** – Splits the string into a list of substrings along the given separator. The separator is not a regular expression.
- **Example:**
  'aaa,bbb,ccc'.split(',') -> ['aaa', 'bbb', 'ccc'].
  If you just write s.split(), it splits the string along the whitespace characters.
- **s.join(list)** The opposite of split(). It joins the elements of a list with a given separator (this becomes the string s).
- **Example:**
  '---'.join(['aaa', 'bbb', 'ccc']) -> aaa---bbb---ccc.

Slicing strings

```
> s = "Batman"
> len(s) -> 6
> s[0] -> 'B'
> s[1:4] -> 'atm'
> s[0:4] -> 'Batm'
> s[3:6] -> 'man'
> s[3:]  -> 'man'
> s[:3] -> 'Bat'
> s[:] -> 'Batman'
> s[-1] -> 'n'
> s[-6] -> 'B'
> s[-3:] -> 'man'
> s[:-3] -> 'Bat'
> s[::2] -> 'Bta'
> s[::1]-> 'Batman'
> s[::-1] -> 'namtaB'
```

Practice

- Delete - Spaces/Lowercase letters/Uppercase letters/Digits/Vowels/Consonats
- Duplicate - Spaces/Lowercase letters/Uppercase letters/Digits/Vowels/Consonats

Homework 1

- Write a program that determines and prints to the standard output how many positive even values were present at the standard input.
- The first line of the input contains a positive integer (n), which is the number of rows containing integers to be processed. Each of the next n rows contains exactly one integer.
- The program should write exactly one integer to the output, which is satisfy the condition.
- **Input:**
  - 5
  - -2
  - -1
  - 0
  - 1
  - 2
- **Output:**
  - 1

Homework 2

- Write a program that reads words up to (end-of-file) *EOF* and removes/deletes the vowels from the input words.

- Input:
  - apple
  - pear
  - peach

- Output:
  - ppl
  - pr
  - pch

Homework 3

- Write a program that reads words up to the *"END"* string, and doubles/duplicates the uppercase letters.

- Input:
  - Apple
  - PeaR
  - PeaCh
  - END

- Output:
  - AApple
  - PPeaRR
  - PPeaCCh