2023. 2학기

웹/파이썬프로그래밍 01
중간고사

담당 교수: 박상근

- 밑줄학번.py 형식으로 파일을 생성하세요 (for example, _2023123456.py), otherwise -1P.
- 제출 전에 print 함수 삭제 또는 주석처리하세요, otherwise -1P.
- input 함수 사용하지 마세요, otherwise -1P.
- 해당 파일을 import 할 때 문제가 발생하면, you get -1P.
- 오타 조심!

시험 파일을 제출 하고 더 제출할 필요가 없다고 판단되면, 노트북을 덮고 종이로 된 시험 자료로 다른 과목 시험을 공부할 수 있습니다. 단, 태블릿이나 스마트폰 등의 디지털기기는 절대 불가!

***str* Method**	**Description**
isalnum()	Returns True if all characters in the string are alphanumeric
isalpha()	Returns True if all characters in the string are in the alphabet
isdigit()	Returns True if all characters in the string are digits
islower()	Returns True if all characters in the string are lower case
isupper()	Returns True if all characters in the string are upper case
lower()	Converts a string into lower case
upper()	Converts a string into upper case
replace()	Returns a string where a specified value is replaced with a specified value
split()	Splits the string at the specified separator, and returns a list

Define the following 10 functions

function1 (1 point)

- The function name: **function1**
- This function takes two positional arguments
  - First argument: **int type**
  - Second argument: **int type**
- The return value: **float type**
  - 두 인자의 합을 float type으로 리턴
- This function will be tested with positional arguments, for example:
  - `print(function1(1, 2))  # 3.0`
  - `print(function1(0, -1))  # -1.0`
  - `print(function1(10, 20))  # 30.0`
  - `print(function1(20, 0))  # 20.0`

function2 (1 point)

- The function name: **function2**
- This function takes a positional arguments
  - Only one argument: **str type (**문자열의 길이는 **1** 이상**)**
- The return value: **int type**
  - 인자로 주어진 문자열 중에서 짝수인 숫자형 문자의 합
  - 짝수인 숫자형 문자가 없다면 0을 리턴
- This function will be tested with positional arguments, for example:
  - print(function2("AB1234CD")) # 6 (because 2+4)
  - print(function2("THREE")) # 0 (because there are no even digits)
  - print(function2("khu9892")) # 10 (because 8+2)
  - print(function2("1")) # 0 (because there are no even digits)
  - print(function2("2460")) # 12 (because 2+4+6+0)

function3 (1 point)

- The function name: **function3**
- This function takes two positional arguments
  - First argument: **str type (**문자열의 길이는 **1** 이상**)**
  - Second argument: **str type (**문자열의 길이는 **1** 이상**)**
- The return value: **bool type**
  - 두 문자열이 대소문자와 상관없이 같은 문자열이라면 True 리턴
  - 그렇지 않다면 False 리턴
- This function will be tested with positional arguments, for example:
  - print(function3("Hell", "HELL")) # True
  - print(function3("ABC", "abc")) # True
  - print(function3("ABC", "abcde")) # False
  - print(function3("Name", "nAME")) # True
  - print(function3("GOOD", "GOLD")) # False

function4 (1 points)

- The function name: **function4**
- This function takes a positional argument
  - Only one argument: **list type (**리스트 안에는 **int** 타입의 값만 존재할 수 있음**)**
- The return value: **int type**
  - 주어진 리스트에서 두 번째로 큰 숫자를 리턴
  - 인자의 길이가 2보다 작다면 -1 리턴
- This function will be tested with positional arguments, for example:
  - print(function4([1, 5, 2])) # 2
  - print(function4([1, 20, 10, 2, 8])) # 10
  - print(function4([7, 9])) # 7
  - print(function4([ ])) # -1 (because a length of the argument is less than 2)
  - print(function4([1])) # -1 (because a length of the argument is less than 2)

function5 (1 points)

- The function name: **function5**
- This function takes one positional arguments: **int type (1**이상의 정수**)**
- The return value: **bool type**
  - 주어진 숫자가 팰린드롬 수라면 True, 아니라면 False 리턴
    (*팰린드롬 수란, 거꾸로 읽어도 똑같은 수를 의미)
- This function will be tested with positional arguments, for example:
  - print(function5(12321)) # True
  - print(function5(987789)) # True
  - print(function5(121212)) # False
  - print(function5(999909)) # False
  - print(function5(1)) # True

function6 (1 point)

- The function name: **function6**
- This function takes two positional arguments
  - The first argument: **int type** (9999보다 큰 수)
  - The second argument: **int type** (네 자리 수, 예: 1000~9999)
- The return value: **bool type**
  - 두 번째 인자가, 첫 번째 인자의 첫 부분 또는 끝 부분에서 딱 일치하면 True 리턴
  - 그렇지 않으면 False 리턴
- This function will be tested with positional arguments, for example:
  - print(function6(12349992, 1234)) # True (because **1234**9992)
  - print(function6(12349992, 9992)) # True (because 1234**9992**)
  - print(function6(71234562, 2345)) # False (because 71**2345**62)
  - print(function6(88376, 1000)) # False (because cannot find **1000** from 88376)
  - print(function6(111111, 1111)) # True (because **1111**11 or 11**1111**)
  - print(function6(111211, 1111)) # False (because cannot find **1111** from 111211)

function7 (1 points)

- The function name: **function7**
- This function takes two positional arguments
  - The first argument: **set type**
    - 집합 안의 모든 값은 **1**이상 **10**이하의 정수
    - 집합 안에 최소 **1**개의 값은 존재
  - The second argument: **set type**
    - 집합 안의 모든 값은 **1**이상 **10**이하의 정수
    - 집합 안에 최소 **1**개의 값은 존재
- The return value: **int type**
  - 두 집합 모두에 속하는 것이 아니라 둘 중 하나의 집합에만 속하는 값의 합 리턴
  - 하나의 집합에만 속하는 값이 없다면 0 리턴
- This function will be tested with positional arguments, for example:
  - print(function7({1, 2}, {3, 4})) # 10 (because 1+2+3+4)
  - print(function7({1, 2, 3}, {2, 3, 4})) # 5 (because 1+4, 2 and 3 are excluded)
  - print(function7({1}, {2, 3})) # 6 (because 1+2+3)
  - print(function7({7, 9}, {9, 7})) # 0 (because all items are excluded)
  - print(function7({1, 2}, {10})) # 13 (because 1+2+10)

function8 (1 points)

- The function name: **function8**
- This function takes one positional argument: **list type (**리스트 안에는 **int** 타입만 존재**)**
  - 리스트에는 최소 하나 이상의 값이 있음
  - 리스트에 한 번만 등장하는 값이 단 하나 존재함. 나머지는 없거나 두 번씩 등장.
- The return value: **int type**
  - 한 번만 등장하는 값을 찾아서 리턴 (한 번만 등장하는 값이 반드시 존재함)
- This function will be tested with positional arguments, for example:
  - print(function8([3,3,2,1,2])) # 1 (because 1 appears once)
  - print(function8([1])) # 1 (because 1 appears once)
  - print(function8([2, -2, 2])) # -2 (because -2 appears once)
  - print(function8([-1, 1, -1, 99, 1, 2, 2])) # 99 (because 99 appears once)

function9 (1 points)

- The function name: **function9**
- This function takes the unknown number of positional arguments: **int types**
- The return value: **int type**
  - 모든 인자(arguments)의 합계 리턴
  - 만약 인자(arguments)가 아무것도 없다면 -1 리턴
- This function will be tested with positional arguments, for example:
  - print(function9(1, 2)) # 3 (because 1+2)
  - print(function9()) # -1 (because there is no argument)
  - print(function9(-9, -1, 10)) # 0 (because (-9)+(-1)+10 is 0)
  - print(function9(10, -30)) # -20 (because 10-30 is -20)
  - print(function9(40)) # 40 (because 40 is 40)

function10 (1 points)

- The function name: **function10**
- This function takes the unknown number of positional or keyword arguments.
  - All arguments are **int types** (greater than 0)
- The return value: **int type**
  - 키워드 인자(keyword arguments)의 값 중 가장 큰 값을 리턴.
  - 키워드 인자(keyword arguments)가 없다면 0을 리턴.
- This function will be tested with positional arguments, for example:
  - print(function10(1, 2, 3, a=3, b=4)) # 4 (4 is the highest among 3 and 4)
  - print(function10(3, n=10, k=20)) # 20 (20 is the highest among 10 and 20)
  - print(function10(height=175)) # 175 (because 175 is the highest value)
  - print(function10(1, 2, 9, 3)) # 0 (because there is no keyword argument)
  - print(function10()) # 0 (because there is no keyword argument)

2023. Fall

Web/Python Programming 01
Mid-term exam

Professor: Sangkeun Park

- Name your Python file, _StudentNumber.py (for example, _2023123456.py), otherwise -1P.
- Remove or comment all ***print*** functions, otherwise -1P.
- Do not use the ***input*** function, otherwise -1P.
- If I fail to import your module due to your code, you get -1P.
- Be careful of typos (see function names).

After submitting the file, if you think you finished the exam, close your laptop (You cannot use digital devices in the class anymore.).
Then you can study for another exam using paper-based materials.

***str* Method**	**Description**
isalnum()	Returns True if all characters in the string are alphanumeric
isalpha()	Returns True if all characters in the string are in the alphabet
isdigit()	Returns True if all characters in the string are digits
islower()	Returns True if all characters in the string are lower case
isupper()	Returns True if all characters in the string are upper case
lower()	Converts a string into lower case
upper()	Converts a string into upper case
replace()	Returns a string where a specified value is replaced with a specified value
split()	Splits the string at the specified separator, and returns a list

Define the following 10 functions

function1 (1 point)

- The function name: **function1**
- This function takes two positional arguments
  - First argument: **int type**
  - Second argument: **int type**
- The return value: **float type**
  - Sum of the two arguments as float type
- This function will be tested with positional arguments, for example:
  - `print(function1(1, 2))  # 3.0`
  - `print(function1(0, -1))  # -1.0`
  - `print(function1(10, 20))  # 30.0`
  - `print(function1(20, 0))  # 20.0`

function2 (1 point)

- The function name: **function2**
- This function takes a positional arguments
  - Only one argument: **str type (a length of the argument is greater than 0)**
- The return value: **int type**
  - Sum of all the even digits from the given string.
  - Return 0, if there are no even digits.
- This function will be tested with positional arguments, for example:
  - `print(function2("AB1234CD"))  # 6 (because 2+4)`
  - `print(function2("THREE"))  # 0 (because there are no even digits)`
  - `print(function2("khu9892"))  # 10 (because 8+2)`
  - `print(function2("1"))  # 0 (because there are no even digits)`
  - `print(function2("2460"))  # 12 (because 2+4+6+0)`

function3 (1 point)

- The function name: **function3**
- This function takes two positional arguments
  - First argument: **str type (a length of the argument is greater than 0)**
  - Second argument: **str type (a length of the argument is greater than 0)**
- The return value: **bool type**
  - If the arguments are the same disregarding their cases, return True.
  - Return False, otherwise
- This function will be tested with positional arguments, for example:
  - `print(function3("Hell", "HELL"))  # True`
  - `print(function3("ABC", "abc"))  # True`
  - `print(function3("ABC", "abcde"))  # False`
  - `print(function3("Name", "nAME"))  # True`
  - `print(function3("GOOD", "GOLD"))  # False`

function4 (1 points)

- The function name: **function4**
- This function takes a positional argument
  - Only one argument: **list type (All items are int type values in the list)**
- The return value: **int type**
  - Return the second biggest number from the first argument.
  - If the length of the first argument is less than 2, return -1.
- This function will be tested with positional arguments, for example:
  - `print(function4([1, 5, 2]))  # 2`
  - `print(function4([1, 20, 10, 2, 8]))  # 10`
  - `print(function4([7, 9]))  # 7`
  - `print(function4([ ]))  # -1 (because a length of the argument is less than 2)`
  - `print(function4([1]))  # -1 (because a length of the argument is less than 2)`

function5 (1 points)

- The function name: **function5**
- This function takes one positional arguments: **int type (greater than 0)**
- The return value: **bool type**
  - If the first argument is a palindromic number, return True. Otherwise, False (*A palindromic number is a number that remains the same when its digits are reversed.)
- This function will be tested with positional arguments, for example:
  - print(function5(12321)) # True
  - print(function5(987789)) # True
  - print(function5(121212)) # False
  - print(function5(999909)) # False
  - print(function5(1)) # True

function6 (1 point)

- The function name: **function6**
- This function takes two positional arguments
  - The first argument: **int type** (greater than 9999)
  - The second argument: **int type** (4-digit numbers, such as 1000~9999)
- The return value: **bool type**
  - If the second argument starts at the beginning of the first argument, return True
  - Or, if the second argument ends at the end of the first argument, return True
  - In all other cases, return False.
- This function will be tested with positional arguments, for example:
  - print(function6(12349992, 1234)) # True (because **1234**9992)
  - print(function6(71234562, 2345)) # False (because 71**2345**62)
  - print(function6(88376, 1000)) # False (because cannot find **1000** from 88376)
  - print(function6(111111, 1111)) # True (because **1111**11 or 11**1111**)
  - print(function6(111211, 1111)) # False (because cannot find **1111** from 111211)

function7 (1 points)

- The function name: **function7**
- This function takes two positional arguments
  - The first argument: **set type**
    - **all items are int types in the set; all items are from 1 to 10.**
    - **at least one item exists**
  - The second argument: **set type**
    - **all items are int types in the set; all items are from 1 to 10.**
    - **at least one item exists**
- The return value: **int type**
  - The sum of items that are present in one of the sets, not presented in both sets.
  - If there is no item presented in only a set, return 0.
- This function will be tested with positional arguments, for example:
  - print(function7({1, 2}, {3, 4})) # 10 (because 1+2+3+4)
  - print(function7({1, 2, 3}, {2, 3, 4})) # 5 (because 1+4, 2 and 3 are excluded)
  - print(function7({1}, {2, 3})) # 6 (because 1+2+3)
  - print(function7({7, 9}, {9, 7})) # 0 (because all items are excluded)
  - print(function7({1, 2}, {10})) # 13 (because 1+2+10)

function8 (1 points)

- The function name: **function8**
- This function takes one positional argument: **list type (all items are int types)**
  - There is at least one item in the list
  - Only one number appears once, but the other numbers appear twice.
- The return value: **int type**
  - Find the number that appears once. (The number that appears once must exist.)
- This function will be tested with positional arguments, for example:
  - `print(function8([3,3,2,1,2]))  # 1 (because 1 appears once)`
  - `print(function8([1]))  # 1 (because 1 appears once)`
  - `print(function8([2, -2, 2]))  # -2 (because -2 appears once)`
  - `print(function8([-1, 1, -1, 99, 1, 2, 2]))  # 99 (because 99 appears once)`

function9 (1 points)

- The function name: **function9**
- This function takes the unknown number of positional arguments: **int types**
- The return value: **int type**
  - Sum of all arguments
  - If there is no argument, return -1
- This function will be tested with positional arguments, for example:
  - `print(function9(1, 2))  # 3 (because 1+2)`
  - `print(function9())  # -1 (because there is no argument)`
  - `print(function9(-9, -1, 10))  # 0 (because (-9)+(-1)+10 is 0)`
  - `print(function9(10, -30))  # -20 (because 10-30 is -20)`
  - `print(function9(40))  # 40 (because 40 is 40)`

function10 (1 points)

- The function name: **function10**
- This function takes the unknown number of positional or keyword arguments.
  - All arguments are **int types** (greater than 0)
- The return value: **int type**
  - The greatest keyword argument value.
  - If there is no keyword argument, the greatest keyword argument value is 0.
- This function will be tested with positional arguments, for example:
  - `print(function10(1, 2, 3, a=3, b=4))  # 4 (4 is the highest among 3 and 4)`
  - `print(function10(3, n=10, k=20))  # 20 (20 is the highest among 10 and 20)`
  - `print(function10(height=175))  # 175 (because 175 is the highest value)`
  - `print(function10(1, 2, 9, 3))  # 0 (because there is no keyword argument)`
  - `print(function10())  # 0 (because there is no keyword argument)`