INF211

LABORATORY LEAFLET

FOR STUDENTS

LABORATORY-4
Function and List Usage

Tasks	Explanations
Exercise 1	Pick
Exercise 2	Remove
Exercise 3	Sort
Task 1	Withdraw
Task 2	Transfer
Task 3	Finding Duplicated String

Exercise 1 - Pick: Write a function that takes two arguments: a list and an index. Return the indexed element from the list. Call the function you wrote with various parameters and print its output to the screen.

- **The index starts from 0.**
- **If the index is < 0 or ≥ length of the list return None.**
- **Function parameter names do not matter, but the function should expect 2 parameters.**
- *inputs:*
  - *l = { x : x is list and len(x) ∈ [0:200]} and $l_{idx} \in \mathbb{Z}$*
  - *idx ∈ ℤ*
- *output: l_{idx} or NoneType*

```
>>> Exercise1_Pick([20, 13, 18], 0)
20
>>> Exercise1_Pick ([20, 13, 18], 2)
18
>>> Exercise1_Pick ([20, 13, 18], -1)
None
>>> Exercise1_Pick ([20, 13, 18], 4)
None
```

Exercise 2 – Remove: Write a function that takes two arguments; a list and an index and return the list with the given indexed element removed from the list. Call the function you wrote with various parameters and print its output to the screen.

- **The index starts from 0.**
- **If the index is < 0 or > length of the list return the list as is.**
- **Function parameter names do not matter, but the function should expect 2 parameters.**
- *inputs:*
  - *l = { x : x is list and len(x) ∈ [0:200]} and $l_i \in \mathbb{Z}$*
  - *idx ∈ ℤ*
- *output: list*

```
>>> Exercise2_Remove([], 2)
[]
>>> Exercise2_Remove([1, 2, 3], 2)
[1, 2]
>>> Exercise2_Remove([1, 2, 3], -1)
[1, 2, 3]
>>> Exercise2_Remove([1, 2, 3], 4)
[1, 2, 3]
```

Exercise 3-Sort: Write a function that takes two arguments; a list of numbers and a boolean. If the boolean value is True, return the list sorted in ascending order, if the boolean value is False, return the list sorted in descending order. Call the function you wrote with various parameters and print its output to the screen.

- **Function parameter names do not matter, but the function should expect 2 parameters.**
- *inputs:*
- ○ *l = { x : x is list and len(x) ∈ [0:200]} and li ∈ ℤ*
  ○ *des ∈ {True, False}*
- *output: list*

```
>>> Exercise3_Sort([2, 1, 3], False)
[3, 2, 1]
>>> Exercise3_Sort([2, 1, 3], True)
[1, 2, 3]
```

Task 1 - Withdraw: Write a function that will withdraw money from a given account and return the updated accounts. The function should take four named arguments:

- `accounts`: a list that will hold the accounts
- `source`: the source index which the money will be withdrawn from
- `lira`: the amount that will hold the lira part of the withdraw
- `kurus`: the amount that will hold the kurus part of the withdraw

Accounts will be a list of strings that will hold all the accounts of the customers. The strings will be in the form of "`lira`". "`kurus`" and you need to make appropriate conversions to do any operations on these strings. The final accounts list that is returned will also be in the same form. An example accounts list would be `["11.23", "10.43", "100.63", "0.10"]`.

- **The index starts from 0.**
- **If the source is < 0 or > length of the `list` return the accounts with no change.**
- **If the transition is not doable (i.e. account has less money than the requested amount) return the accounts with no change.**
- **lira part of the string should NOT have any leading 0's. (ie. should be "2" instead of "02")**
- **kurus part of the string should have the preceding 0's unless it is actually 0. If it is 0, just keep 0. (ie. should be ".20" instead of ".2", ".40" instead of ".4", and ".0" instead of ".00")**
- **Function parameter names should be `accounts`, `source`, `lira`, `kurus`.**
- *inputs :*
  - *accounts = { x : x is list of strings and len(x) ∈ [1:200]}*
  - *source ∈ ℤ*
  - *lira = { x : x ∈ [0:1E6]}*
  - *kurus = { x : x ∈ [0:99]}*
- *outputs : accounts = { x : x is list and len(x) ∈ [1:200]}*

```
>>> Task1_Withdraw (accounts=["11.23", "10.43", "100.63", "0.10"], source=-1, lira=10, kurus=13)

["11.23", "10.43", "100.63", "0.10"]

>>> Task1_Withdraw (["11.23", "10.43", "100.63", "0.10"], 1, 10, 13)

["11.23", "0.30", "100.63", "0.10"]

>>> Task1_Withdraw (["11.23", "10.43", "100.63", "0.10"], 4, 10, 13)

["11.23", "10.43", "100.63", "0.10"]

>>> Task1_Withdraw (["11.23", "10.43", "100.63", "0.10"], 3, 10, 13)

["11.23", "10.43", "100.63", "0.10"]
```

Task 2-Transfer: Write a function that will transfer money between given source and destination accounts and return the updated accounts. The function should take six named arguments:

- `accounts`: a list of strings that will hold the money on all the accounts
- `source`: the source index which the money will be transferring from
- `destination`: the destination index which the money will be transferring to
- `lira`: the amount that will hold the lira part of the transfer
- `kurus`: the amount that will hold the kurus part of the transfer
- `fee`: True or False. Should default to False. This will be used to apply transfer fees to the transaction or not.

Accounts will be a list of strings that will hold all the accounts of the customers. The strings will be in the form of "`lira`". "`kurus`" and you need to make appropriate conversions to do any operations on these strings. The final accounts list that is returned will also be in the same form. An example accounts list would be `["11.23", "10.43", "100.63", "0.10"]`.

- **Index starts from 0.**
- **The transfer amount should be subtracted from the source, and added to the destination.**
  - **If the fee parameter is set to True, the transfer fee should be subtracted from the source.**
- **The transfer fee should be 0.1 for any amount less than 10.0 liras, for any amount that is greater than 10.0 liras, the transfer fee should be 1% of the transfer amount.**
  - **Careful with floating point operations here. (i.e. 1% of 115 should be 1.15, and not 1.1500000000000001). Take the first two digits after the dot. (i.e. 0.1013 will be 0.10)**
- **If the source and the destination indexes are the same, return the accounts list with no change.**
- **If the source or the destination are less than 0 or greater than the length of the list return the accounts list with no change.**
- **If the transition is not doable (i.e. account has less money than the requested amount) return the accounts list with no change.**
  - **If the fee parameter is set to True, and if the transition + the transfer fee is not doable (i.e. account has less money than the requested amount + the transfer fee) return the accounts with no change.**
- `lira` **part of the string should NOT have any leading 0's. (ie. should be "2" instead of "02")**
- `kurus` **part of the string should have the preceding 0's unless it is actually 0. If it is 0, just keep 0. (ie. should be ".20" instead of ".2", ".40" instead of ".4", and ".0" instead of ".00")**
- **Function parameter names should be** `accounts`, `source`, `destination`, `lira`, `kurus`, `fee`.
- *inputs :*
  - *accounts = { x : x is list of strings and len(x) ∈ [1:200]}*
  - *source, destination ∈ ℤ*
  - *lira = { x : x ∈ [0:1E6]}*
  - *kurus = { x : x ∈ [0:99]}*
  - *fee ∈ {True, False}*
- *outputs : accounts = { x : x is list and len(x) ∈ [1:200]}*

```
>>> Task2_Transfer (accounts=["11.23", "10.43", "100.63", "0.10"], source=0, destination=1, lira=10, kurus=13)
```

```
>>> Task2_Transfer (accounts=["11.23", "10.43", "100.63", "0.10"],
source=0, destination=0, lira=10, kurus=13)

["11.23", "10.43", "100.63", "0.10"]

>>> Task2_Transfer (accounts=["11.23", "10.43", "100.63", "0.10"],
source=3, destination=0, lira=10, kurus=13)

["11.23", "10.43", "100.63", "0.10"]

>>> Task2_Transfer (accounts=["11.23", "10.43", "100.63", "0.10"],
source=0, destination=1, lira=9, kurus=13, fee=True)

["2.0", "19.56", "100.63", "0.10"]

>>> Task2_Transfer (accounts=["11.23", "10.43", "100.63", "0.10"],
source=0, destination=1, lira=10, kurus=13, fee=True)

["1.0", "20.56", "100.63", "0.10"]

Explanation: %1 of 10.13 is 0.1013. Taking the first two digits after
dot gives: 0.10.
```

Task 3- Finding Duplicated String: Write a function that will find and return the only string that has a duplicate in a given list.

- **Function parameter name does not matter, but function should expect 1 parameter.**
- **Return None if there is no duplicate.**
- *inputs : a = { x : x is list of strings and len(x) ∈ [0:200]}*
- *source ∈ ℤ*
- *output : string*

```
>>> Task3_FindDuplicated (['1', '25', '36', '4', '1', '5'])
1
>>>Task3_FindDuplicated(['a','s','blasdf','dx','c','x','z','1','c','5'])
c
>>> Task3_FindDuplicated (['aba', 'hede', '1', 'aba'])
aba
>>> Task3_FindDuplicated ([])
None
>>> Task3_FindDuplicated (['aba'])
None
```