МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»
ІКТА
Кафедра ЗІ

Звіт

до лабораторної роботи №7

з курсу: «Програмування скриптовими мовами» на тему: «Створення та використання функцій»

Виконав:
студент групи КБ-305
Семенчук А.А.
Прийняв:
к.т.н., доцент
Совин Я. Р.

Львів 2024

Мета роботи – ознайомитись з можливостями і застосуванням функцій у мові Python.

Завдання

1. Встановити з допомогою pip статичний аналізатор типів mypy. Для mypy правила перевірки задаються у файлі mypy.ini, який розташуйте разом з файлом test_mypy.py. Файли test_mypy.py та mypy.ini є в матеріалах до лабораторної роботи.

2. Переглянути файл test_mypy.py, самостійно відзначити порушення правил анотацій типів в ньому.

3. Скоригуйте знайдені порушення та збережіть виправлений файл як test_mypy_my.py.

4. Перевірте файл test_mypy.py з допомогою mypy і порівняйте їх результати зі своїми.

5. Перевірте файл test_mypy_my.py з допомогою mypy і оцініть наскільки добре ви провели виправлення. Усуньте всі помилки і зауваження, які видав чекер.

6. Написати програму валідації введеного паролю з Лабораторної роботи №4 з використанням функцій. Користувач вводить пароль, програма має перевірити наявність у ньому лише заданих типів символів у вказаних пропорціях і з дотримання додаткових правил згідно варіанту у табл. 1 і 32 вивести інформацію про результати перевірки у форматі як показано на рис.

7. Написати програму генерації паролю. з Лабораторної роботи №3 з використанням функцій. Користувач повинен ввести кількість різних типів символів у паролі згідно варіанту у табл. 2 і вивести згенерований пароль у форматі як показано на рис

8. Написати програму яка створює і виводить двовимірний список з 5 елементів. Кожен елемент списку представляє собою список, який містить опис атрибутів об’єкту згідно таблиці 3. Організуйте діалоговий режим із вводом з клавіатури, який дозволяє робити такі операції:

a. Вивести весь список.

b. Додавати елементи до списку.

c. Відсортувати список за заданим атрибутом.

d. Видаляти елементи за заданим атрибутом.

e. Видаляти елемент за заданим індексом.

f. Виводити всі елементи за заданим атрибутом. Всі операції для пунктів 6-8 повинні бути оформлені у вигляді функцій з анотаціями і проходити перевірку mypy. Номер варіанту відповідає номеру в списку групи

Виконання лабораторної роботи

1.

```
@Velovo123 ➜/workspaces/python-labs/lab7 (main) $ pip install mypy
Collecting mypy
  Downloading mypy-1.13.0-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.manylinux_2_28_x86_64.whl.metadata (2.1 kB)
Requirement already satisfied: typing-extensions>=4.6.0 in /home/codespace/.local/lib/python3.12/site-packages (from mypy) (4.9.0)
Collecting mypy-extensions>=1.0.0 (from mypy)
  Downloading mypy_extensions-1.0.0-py3-none-any.whl.metadata (1.1 kB)
Downloading mypy-1.13.0-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.manylinux_2_28_x86_64.whl (12.6 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 12.6/12.6 MB 57.9 MB/s eta 0:00:00
Downloading mypy_extensions-1.0.0-py3-none-any.whl (4.7 kB)
Installing collected packages: mypy-extensions, mypy
Successfully installed mypy-1.13.0 mypy-extensions-1.0.0
```

2.

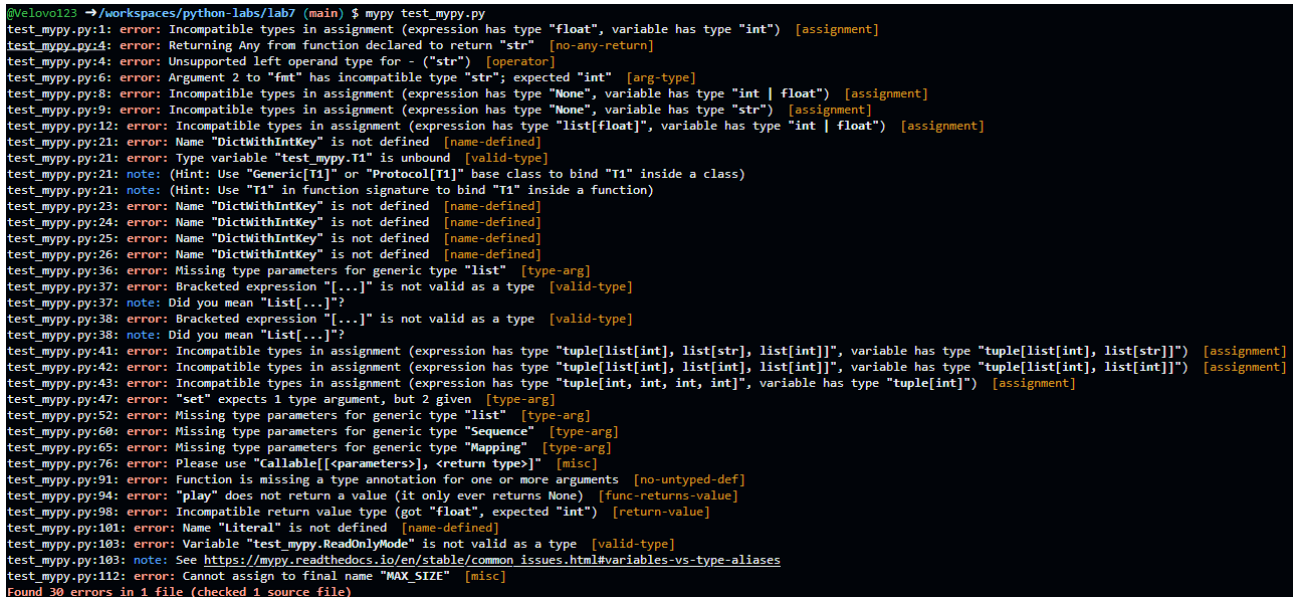
test_mypy.py

```python
variable_name: int = 10 / 2


def fmt(value: str, excitement: int = 10) -> str:
    return value - "!"*excitement

fmt('Hello', 'word!!!')

my_age: int | float = None
my_city: str = None

number = int | float
prices: number = [100, 105, 125.5]

Image = list[list[int]]
image: Image = [*range(1000)]


from typing import TypeVar

T1 = TypeVar('T1')
DictWithIntKey[T1] = dict[int, T1]

a: DictWithIntKey[str] = {0: 'zero', 1: 'one'}
b: DictWithIntKey[bool] = {0: False, 1: True}
c: DictWithIntKey[int] = {0: 0, 1: 1}
d: DictWithIntKey[float] = {0: 0.0, 1: 1.0}

T2 = TypeVar('T2')
def combine(a: T2, b: T2) -> str:
    return str(a) + str(b)

print(combine(10, 20))
print(combine('hello', 2020))
print(combine('hello', 'word'))

lst_1: list = [1, "2"]
lst_2: [int] = [1, 2, 3, 4]
lst_3: list[[int]] = [[1, 2], [3, 4]]

tpl_1: tuple[int, ...] = 1, 2, 3, 4
tpl_2: tuple[list[int], list[str]] = ([1, 2], ['1', '2'], [1, 2])
tpl_3: tuple[list[int], list[int]] = ([1, 2], [3, 4], [5, 6])
tpl_4: tuple[int] = (1, 2, 3, 4)

dt_1: dict[str, int] = {"a": 1, "b": 2}

st_1: set[int, str] = {1, 2, '3', '4'}


from typing import Any
```

```python
def get_first(items: list) -> Any:
    return items[0]

get_first(['1', 1])


from typing import Sequence, Mapping

def first_element(items: Sequence) -> Any:
    return items[0]

first_element((1, 2, 3))

def get_value(data: Mapping, key: str) -> Any:
    return data.get(key)

get_value({'1': 2.0, '3': 4.0}, key = '1')


from typing import Callable

def is_twice_as_big(num1: int, num2: int) -> bool:
    return num1 >= 2 * num2

def compare_nums(num1: int, num2: int, comp: Callable[int, int, bool]) -> int:
    if comp(num1, num2):
        return num1
    else:
        return num2

compare_nums(10, 3, is_twice_as_big)


from typing import Generator

def do_twice() -> Generator[int]:
    yield 1
    yield 2

def play(player_name) -> None:
    print(f"Хід {player_name}")

ret_val = play("Іван")


def calc_div(a: int, b: int) -> int:
    return a / b


ReadOnlyMode = Literal["r", "r+"]

def read_file(file_name: str, mode: ReadOnlyMode) -> None:
    return
```

```python
read_file('data.txt', 'w')


from typing import Final

MAX_SIZE: Final = 1_000
MAX_SIZE += 1
```

4.



3.

test_mypy_my.py

```python
from typing import TypeVar, Any, Sequence, Mapping, Callable, Generator, Final,
Literal, Union, List, Dict, Tuple, Set


# Corrected variable assignment with integer type
variable_name: int = int(10 / 2)

# Corrected function to use addition instead of unsupported subtraction with
strings
def fmt(value: str, excitement: int = 10) -> str:
    return value + "!" * excitement

# Fixed argument to match expected type
fmt('Hello', 10)

# Corrected type to be compatible with possible None values
my_age: Union[int, float, None] = None
my_city: Union[str, None] = None

# Changed number to a valid type alias and corrected prices to a List
```

```python
Number = Union[int, float]
prices: List[Number] = [100, 105, 125.5]

# Corrected typing for the image variable
Image = List[List[int]]
image: Image = [[0] * 1000]

# Typing corrected for dictionary with generic int key and any type
T1 = TypeVar('T1')
DictWithIntKey = Dict[int, T1]

a: DictWithIntKey[str] = {0: 'zero', 1: 'one'}
b: DictWithIntKey[bool] = {0: False, 1: True}
c: DictWithIntKey[int] = {0: 0, 1: 1}
d: DictWithIntKey[float] = {0: 0.0, 1: 1.0}

# Fixed type annotations for function parameters
T2 = TypeVar('T2')
def combine(a: T2, b: T2) -> str:
    return str(a) + str(b)

# Test calls to combine function with matching types
print(combine(10, 20))
print(combine('hello', 'world'))

# Corrected list and tuple annotations
lst_1: List[Union[int, str]] = [1, "2"]
lst_2: List[int] = [1, 2, 3, 4]
lst_3: List[List[int]] = [[1, 2], [3, 4]]

tpl_1: Tuple[int, ...] = (1, 2, 3, 4)
tpl_2: Tuple[List[int], List[str]] = ([1, 2], ['1', '2'])
tpl_3: Tuple[List[int], List[int]] = ([1, 2], [3, 4])
tpl_4: Tuple[int, ...] = (1, 2, 3, 4)

# Corrected dict typing
dt_1: Dict[str, int] = {"a": 1, "b": 2}

# Fixed set annotation
st_1: Set[Union[int, str]] = {1, 2, '3', '4'}

# Corrected type annotations in functions
def get_first(items: List[Any]) -> Any:
    return items[0]

get_first(['1', 1])

def first_element(items: Sequence[Any]) -> Any:
    return items[0]

first_element((1, 2, 3))

def get_value(data: Mapping[str, Any], key: str) -> Any:
```

```python
    return data.get(key)

get_value({'1': 2.0, '3': 4.0}, key='1')

# Corrected callable type syntax
def is_twice_as_big(num1: int, num2: int) -> bool:
    return num1 >= 2 * num2

def compare_nums(num1: int, num2: int, comp: Callable[[int, int], bool]) -> int:
    if comp(num1, num2):
        return num1
    else:
        return num2

compare_nums(10, 3, is_twice_as_big)

# Added generator type annotation
def do_twice() -> Generator[int, None, None]:
    yield 1
    yield 2

# Corrected return type annotation for play function to match None return type
def play(player_name: str) -> None:
    print(f"Хід {player_name}")

play("Іван")

# Fixed division return type to float
def calc_div(a: int, b: int) -> float:
    return a / b

# Added typing for Literal and corrected function
ReadOnlyMode = Literal["r", "r+"]

def read_file(file_name: str, mode: ReadOnlyMode) -> None:
    pass

read_file('data.txt', 'r')

# Fixed Final usage
MAX_SIZE: Final = 1_000
# MAX_SIZE += 1 # This line is removed because MAX_SIZE is immutable due to Final
type
```

5.

```
@Velovo123 ➜/workspaces/python-labs/lab7 (main) $ mypy test_mypy_my.py
Success: no issues found in 1 source file
```

6.

```python
from colorama import Fore, Style
```

```python
# Набори символів
upp_char = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
low_char = "abcdefghijklmnopqrstuvwxyz"
num_char = "0123456789"
spc_char = "!@#$%^&*_-"

# Функція для перевірки кожного правила
def check_rule(description: str, condition: bool) -> str:
    color = Fore.GREEN if condition else Fore.RED
    status = "OK!" if condition else "FAIL!"
    print(f"{description} - {color}{status}{Style.RESET_ALL}")
    return condition

# Основна функція для валідації пароля
def validate_password(password: str) -> None:
    # Перевірка довжини пароля
    length_check = check_rule("Довжина не менше 10 символів", len(password) >= 10)

    # Перевірка дозволених символів
    allowed_chars_check = check_rule("Пароль містить лише допустимі символи",
                                     all(c in (upp_char + low_char + num_char +
spc_char) for c in password))

    # Підрахунок кількості кожного типу символів
    low_count = sum(1 for c in password if c in low_char)
    upp_count = sum(1 for c in password if c in upp_char)
    num_count = sum(1 for c in password if c in num_char)
    spc_count = sum(1 for c in password if c in spc_char)

    # Валідація кожного правила
    rules = [
        ("Не менше 2 маленьких латинських літер", low_count >= 2),
        ("Не більше 4 маленьких латинських літер", low_count <= 4),
        ("Не менше 2 цифр", num_count >= 2),
        ("Не більше 4 цифр", num_count <= 4),
        ("Не менше 2 спеціальних символів", spc_count >= 2),
        ("Не більше 3 спеціальних символів", spc_count <= 3),
        ("Не менше 2 великих латинських літер", upp_count >= 2),
        ("Не більше 5 великих латинських літер", upp_count <= 5),
        ("Не більше 3 однакових великих латинських літер підряд",
         not any(password[i] == password[i+1] == password[i+2] for i in
range(len(password) - 2) if password[i] in upp_char)),
        ("Не більше 3 однакових спеціальних символів підряд",
         not any(password[i] == password[i+1] == password[i+2] for i in
range(len(password) - 2) if password[i] in spc_char))
    ]

    # Перевірка правил
    results = [check_rule(description, condition) for description, condition in
rules]

    # Перевірка загальної валідності пароля
```

```
    is_valid = all([length_check, allowed_chars_check] + results)
    print(Fore.GREEN + "Пароль валідний!" + Style.RESET_ALL if is_valid else
Fore.RED + "Пароль не валідний!" + Style.RESET_ALL)

# Запит на введення пароля
password = input("Введіть пароль довжиною не менше 10 символів: ")
validate_password(password)
```

```
@Velovo123 →/workspaces/python-labs/lab7 (main) $ python validate_password.py
Введіть пароль довжиною не менше 10 символів: asdsadsads
Довжина не менше 10 символів - OK!
Пароль містить лише допустимі символи - OK!
Не менше 2 маленьких латинських літер - OK!
Не більше 4 маленьких латинських літер - FAIL!
Не менше 2 цифр - FAIL!
Не більше 4 цифр - OK!
Не менше 2 спеціальних символів - FAIL!
Не більше 3 спеціальних символів - OK!
Не менше 2 великих латинських літер - FAIL!
Не більше 5 великих латинських літер - OK!
Не більше 3 однакових великих латинських літер підряд - OK!
Не більше 3 однакових спеціальних символів підряд - OK!
Пароль не валідний!
```

7.

```
import random

# Інформація про студента
def display_student_info():
    print("Семенчук Анатолій Анатолійович, КБ-305, 2024. Варіант 27")

# Функція для введення кількості символів
def get_symbol_counts():
    upp_count = int(input("Введіть кількість великих латинських літер в паролі: "))
    low_count = int(input("Введіть кількість малих латинських літер в паролі: "))
    num_count = int(input("Введіть кількість цифр в паролі: "))
    spc_count = int(input("Введіть кількість спеціальних символів !@#$%^&*_- в
паролі: "))
    return upp_count, low_count, num_count, spc_count

# Функція для генерації паролю
def generate_password(upp_count, low_count, num_count, spc_count):
    # Символи для паролю
    low_char = "abcdefghijklmnopqrstuvwxyz"
    upp_char = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
    num_char = "0123456789"
    spc_char = "!@#$%^&*_-"

    # Формування паролю
```

```
    password = (
        random.sample(low_char, low_count) +
        random.sample(num_char, num_count) +
        random.sample(spc_char, spc_count) +
        random.sample(upp_char, upp_count)
    )

    # Перемішування символів і повернення паролю
    random.shuffle(password)
    return "".join(password)

# Основна програма
display_student_info()
upp_count, low_count, num_count, spc_count = get_symbol_counts()
password = generate_password(upp_count, low_count, num_count, spc_count)
print("Password:", password)
```

```
@Velovo123 →/workspaces/python-labs/lab7 (main) $ python generate__password.py
Введіть кількість символів для кожного типу:
Введіть кількість великих латинських літер у паролі: 4
Введіть кількість малих латинських літер у паролі: 1
Введіть кількість цифр у паролі: 1
Введіть кількість спеціальних символів у паролі (!@#$%^&*-_): 2
Згенерований пароль: Kx-P6&NF
```

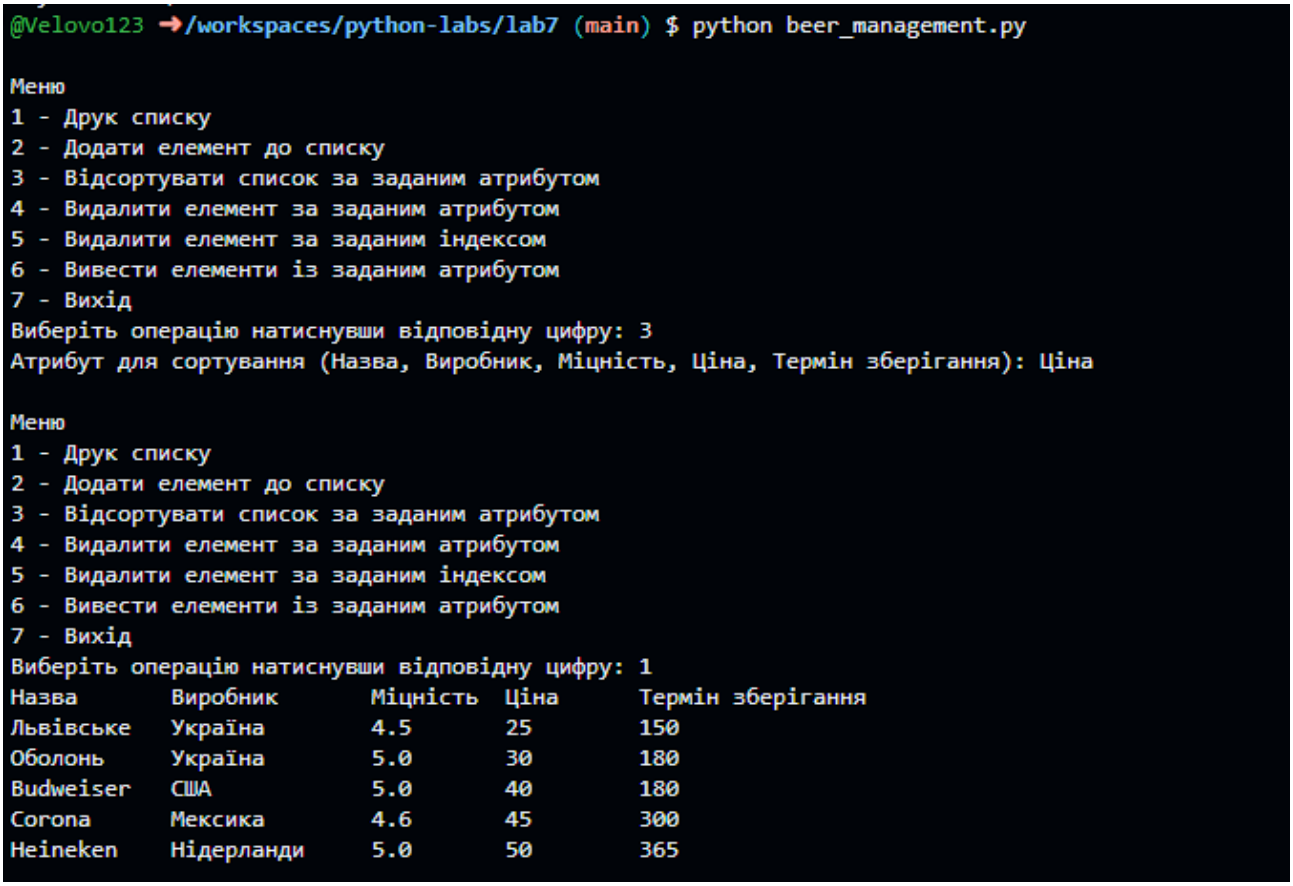
8.

```
from typing import List, Dict, Any

# Створення бази даних (двовимірний список) з інформацією про пиво
beer_database: List[Dict[str, Any]] = [
    {"Назва": "Оболонь", "Виробник": "Україна", "Міцність": 5.0, "Ціна": 30,
"Термін зберігання": 180},
    {"Назва": "Львівське", "Виробник": "Україна", "Міцність": 4.5, "Ціна": 25,
"Термін зберігання": 150},
    {"Назва": "Heineken", "Виробник": "Нідерланди", "Міцність": 5.0, "Ціна": 50,
"Термін зберігання": 365},
    {"Назва": "Corona", "Виробник": "Мексика", "Міцність": 4.6, "Ціна": 45, "Термін
зберігання": 300},
    {"Назва": "Budweiser", "Виробник": "США", "Міцність": 5.0, "Ціна": 40, "Термін
зберігання": 180}
]

# a. Вивести весь список
def print_database(db: List[Dict[str, Any]]) -> None:
    print(f"{'Назва':<12}{'Виробник':<15}{'Міцність':<10}{'Ціна':<10}{'Термін
зберігання':<15}")
    for beer in db:
        print(f"{beer['Назва']:<12}{beer['Виробник']:<15}{beer['Міцність']:<10}{bee
r['Ціна']:<10}{beer['Термін зберігання']:<15}")
```

```python
# b. Додавати елементи до списку
def add_beer(db: List[Dict[str, Any]], name: str, manufacturer: str, strength:
float, price: int, storage: int) -> None:
    db.append({"Назва": name, "Виробник": manufacturer, "Міцність": strength,
"Ціна": price, "Термін зберігання": storage})

# c. Відсортувати список за заданим атрибутом
def sort_database(db: List[Dict[str, Any]], attribute: str) -> None:
    db.sort(key=lambda beer: beer[attribute])

# d. Видаляти елементи за заданим атрибутом
def delete_by_attribute(db: List[Dict[str, Any]], attribute: str, value: Any) ->
None:
    db[:] = [beer for beer in db if beer[attribute] != value]

# e. Видаляти елемент за заданим індексом
def delete_by_index(db: List[Dict[str, Any]], index: int) -> None:
    if 0 <= index < len(db):
        db.pop(index)

# f. Виводити всі елементи за заданим атрибутом
def filter_by_attribute(db: List[Dict[str, Any]], attribute: str, value: Any) ->
None:
    print(f"{'Назва':<12}{'Виробник':<15}{'Міцність':<10}{'Ціна':<10}{'Термін
зберігання':<15}")
    for beer in db:
        if beer[attribute] == value:
            print(f"{beer['Назва']:<12}{beer['Виробник']:<15}{beer['Міцність']:<10}
{beer['Ціна']:<10}{beer['Термін зберігання']:<15}")

# Основне меню
def main() -> None:
    while True:
        print("\nМеню")
        print("1 - Друк списку")
        print("2 - Додати елемент до списку")
        print("3 - Відсортувати список за заданим атрибутом")
        print("4 - Видалити елемент за заданим атрибутом")
        print("5 - Видалити елемент за заданим індексом")
        print("6 - Вивести елементи із заданим атрибутом")
        print("7 - Вихід")

        choice = input("Виберіть операцію натиснувши відповідну цифру: ")
        if choice == "1":
            print_database(beer_database)
        elif choice == "2":
            name = input("Назва: ")
            manufacturer = input("Виробник: ")
            strength = float(input("Міцність: "))
            price = int(input("Ціна: "))
            storage = int(input("Термін зберігання: "))
            add_beer(beer_database, name, manufacturer, strength, price, storage)
```

```
        elif choice == "3":
            attribute = input("Атрибут для сортування (Назва, Виробник, Міцність,
Ціна, Термін зберігання): ")
            sort_database(beer_database, attribute)
        elif choice == "4":
            attribute = input("Атрибут для видалення (Назва, Виробник, Міцність,
Ціна, Термін зберігання): ")
            value = input("Значення атрибуту для видалення: ")
            delete_by_attribute(beer_database, attribute, value)
        elif choice == "5":
            index = int(input("Введіть індекс для видалення: "))
            delete_by_index(beer_database, index)
        elif choice == "6":
            attribute = input("Атрибут для фільтрації (Назва, Виробник, Міцність,
Ціна, Термін зберігання): ")
            value = input("Значення атрибуту для фільтрації: ")
            filter_by_attribute(beer_database, attribute, value)
        elif choice == "7":
            print("Вихід з програми.")
            break
        else:
            print("Невірний вибір. Спробуйте ще раз.")

# Запуск програми
main()
```

```
@Velovo123 ➜/workspaces/python-labs/lab7 (main) $ mypy beer_management.py
Success: no issues found in 1 source file
@Velovo123 ➜/workspaces/python-labs/lab7 (main) $ mypy validate_password.py
validate_password.py:55: error: invalid syntax  [syntax]
Found 1 error in 1 file (errors prevented further checking)
@Velovo123 ➜/workspaces/python-labs/lab7 (main) $ mypy validate_password.py
Success: no issues found in 1 source file
@Velovo123 ➜/workspaces/python-labs/lab7 (main) $ mypy generate__password.py
Success: no issues found in 1 source file
```



В ході роботи було досягнуто мету — ознайомлення з можливостями та застосуванням функцій у мові Python. Ми розглянули основи створення та використання функцій, що дозволяють структурувати код, полегшувати його розуміння і повторне використання. Впровадження функцій дозволило покращити логіку програми, зменшити дублювання коду та підвищити його читабельність. Отримані знання є важливими для побудови більш складних програмних структур, адже функції допомагають організувати код в більш структуровану і зручну для тестування форму.