Звіт з лабораторної роботи No6

Тема роботи: Розробка ігор з умовами

Виконав: Кравченко Сергій, ТЦР-12

Дата: 06.06.2025

Мета роботи:

Хід роботи:

Постановка завдання:

Завдання 1. «Гра "Скарбниця короля" (з обробкою винятків)»

Створіть програму, де гравець повинен поділити знайдені золоті монети

(ціле число від 1 до 1000) між своєю командою (кількість людей вводить

користувач). Використовуйте конструкції try-except-finally:

● Обробіть винятки:

○ некоректний ввід кількості людей (нечислове значення).

○ поділ на нуль.

● Після завершення кожної спроби повідомте користувачу результат дії

або опис помилки.

● У блоці finally завжди виводьте повідомлення: «Пригоди

тривають!».

Код програми:

def kings\_treasure():

print("Гра 'Скарбниця короля'")

print("Ви знайшли скарб з золотими монетами!")

try:

coins = random.randint(1, 1000)

print(f"Ви знайшли {coins} золотих монет!")

people = int(input("На скільки людей потрібно поділити монети? "))

if people == 0:

raise ZeroDivisionError("Не можна ділити на нуль!")

elif people < 0:

raise ValueError("Кількість людей не може бути від'ємною!")

share = coins // people

remainder = coins % people

print(f"Кожен отримує {share} монет. Залишок: {remainder} монет.")

except ValueError as e:

print(f"Помилка: {e} Будь ласка, введіть ціле число.")

except ZeroDivisionError:

print("Помилка: Не можна ділити на нуль!")

finally:

print("Пригоди тривають!")

kings\_treasure()

Приклад роботи програми:

Гра 'Скарбниця короля'

Ви знайшли скарб з золотими монетами!

Ви знайшли 743 золотих монет!

На скільки людей потрібно поділити монети? 12

Кожен отримує 61 монет. Залишок: 11 монет.

Пригоди тривають!

Завдання 2. «Гра "Код сейфу з таймером" (з обробкою винятків)»

Напишіть гру, де гравець має зламати випадково згенерований тризначний

код сейфу за 5 спроб. Реалізуйте таке:

● Використовуйте конструкцію try-except-finally:

перехоплюйте ситуацію, коли користувач вводить нечислове

значення.

● Після кожної спроби (навіть помилкової) виводьте кількість спроб,

що залишились.

● Гравець повинен отримувати підказки: «Код більший» або «Код

менший» після числових спроб.

● Після вичерпання спроб повідомте код сейфу.

Код програми:

def safe\_cracker():

print("Гра 'Код сейфу з таймером'")

secret\_code = random.randint(100, 999)

attempts\_left = 5

try:

while attempts\_left > 0:

print(f"\nЗалишилось спроб: {attempts\_left}")

guess = input("Введіть тризначний код: ")

if not guess.isdigit() or len(guess) != 3:

raise ValueError("Потрібно ввести тризначне число!")

guess = int(guess)

if guess == secret\_code:

print("Вітаємо! Ви зламали сейф!")

return

elif guess < secret\_code:

print("Код більший")

else:

print("Код менший")

attempts\_left -= 1

print(f"\nНа жаль, ви не вгадали. Код сейфу: {secret\_code}")

except ValueError as e:

print(f"Помилка: {e}")

finally:

print("Гра завершена. Дякуємо за гру!")

safe\_cracker()

Приклад роботи програми:

Введіть число від 1 до 100: 50

Число знаходиться в межах від 1 до 100

Введіть число від 1 до 100: 150

Помилковий ввід! Число повинно бути від 1 до 100

Завдання 3: "Складний логічний вираз для відбору кандидатів"

Розробіть програму, яка приймає наступні дані про кандидата: вік,

кількість років досвіду та наявність спеціальної кваліфікації (True/False).

● Сформуйте логічний вираз, який перевіряє, чи кандидат може

претендувати на посаду за такими умовами:

○ Вік має бути не менше 25 років і не більше 50 років,

○ А досвід роботи має бути не менше 3 років або наявність

спеціальної кваліфікації повинна бути True.

● Виведіть повідомлення «Кандидат відповідає вимогам» або

«Кандидат не відповідає вимогам» відповідно до результату

перевірки.

Код програми:

def safe\_cracker():

print("Гра 'Код сейфу з таймером'")

secret\_code = random.randint(100, 999)

attempts\_left = 5

try:

while attempts\_left > 0:

print(f"\nЗалишилось спроб: {attempts\_left}")

guess = input("Введіть тризначний код: ")

if not guess.isdigit() or len(guess) != 3:

raise ValueError("Потрібно ввести тризначне число!")

guess = int(guess)

if guess == secret\_code:

print("Вітаємо! Ви зламали сейф!")

return

elif guess < secret\_code:

print("Код більший")

else:

print("Код менший")

attempts\_left -= 1

print(f"\nНа жаль, ви не вгадали. Код сейфу: {secret\_code}")

except ValueError as e:

print(f"Помилка: {e}")

finally:

print("Гра завершена. Дякуємо за гру!")

safe\_cracker()

Приклад роботи програми:

Гра 'Код сейфу з таймером'

Залишилось спроб: 5

Введіть тризначний код: 500

Код більший

Введіть тризначний код: 600

Код менший

Введіть тризначний код: 550

Код більший

Введіть тризначний код: 580

Код більший

Введіть тризначний код: 582

Вітаємо! Ви зламали сейф!

Гра завершена. Дякуємо за гру!

Завдання 4. «Система нарахування бонусів із множниками»

Напишіть програму, яка симулює ігрову ситуацію: користувач вводить

кількість набраних очок у грі (від 0 до 100). Використовуйте складнішу

систему оцінювання результатів із множниками та рейтингами гравців.

Забезпечте такі умови:

● Якщо кількість очок менше 0 або більше 100, програма повинна

перехопити помилку через try-except та повідомити:

«Некоректне введення! Кількість очок повинна бути в межах від 0 до

100».

● Для результатів від 0 до 49 очок — рейтинг «Початківець», множник

×1 (бонус не нараховується).

● Від 50 до 69 очок — рейтинг «Срібний гравець», множник ×1.5.

● Від 70 до 89 очок — рейтинг «Золотий гравець», множник ×2.

● Від 90 до 100 очок — рейтинг «Платиновий гравець», множник ×3.

Після обчислення фінальної кількості очок (початкові очки × множник)

виведіть результат із повідомленням, наприклад:

Ваш рейтинг: Золотий гравець! Ви отримали 160 балів

(множник ×2)!

Код програми:

def bonus\_system():

print("Система нарахування бонусів")

try:

points = int(input("Введіть кількість набраних очок (0-100): "))

if points < 0 or points > 100:

raise ValueError("Кількість очок повинна бути в межах від 0 до 100")

if points < 50:

rating = "Початківець"

multiplier = 1

elif points < 70:

rating = "Срібний гравець"

multiplier = 1.5

elif points < 90:

rating = "Золотий гравець"

multiplier = 2

else:

rating = "Платиновий гравець"

multiplier = 3

total\_points = int(points \* multiplier)

print(f"\nВаш рейтинг: {rating}! Ви отримали {total\_points} балів (множник ×{multiplier})!")

except ValueError as e:

print(f"Помилка: {e}")

finally:

print("Система завершила роботу.")

bonus\_system()

Приклад роботи програми:

Система нарахування бонусів

Введіть кількість набраних очок (0-100): 78

Ваш рейтинг: Золотий гравець! Ви отримали 156 балів (множник ×2)!

Система завершила роботу.

Завдання 5. «Гра "Втеча з острова піратів" (із комплексною обробкою

винятків)

Реалізуйте текстову гру, де гравець опинився на острові та повинен

вибрати правильний спосіб втечі. Гравець послідовно проходить три етапи:

Етапи гри:

1. Перетин річки:

Гравець вводить кількість деревини для побудови плоту (від 1 до

10).

○ Якщо гравець вводить нечислове значення, перехопіть

помилку та повідомте: «Це не число!» (try-except).

○ Якщо введено число менше 3, повідомте «Деревини замало,

пліт затонув!» і гру завершено.

2. Втеча від піратів:

Гравець обирає спосіб втечі («бігти», «сховатися», «битися»).

Якщо вибір не відповідає жодному з трьох варіантів, перехопіть це

як помилку через try-except, виведіть «Такого варіанту немає,

пірати вас спіймали!», гру завершено.

3. Відкриття скрині зі скарбами:

Гравець повинен ввести секретний код (двозначне число).

○ Використайте try-except для обробки некоректного вводу

(нечислові значення).

○ Якщо код неправильний (не дорівнює попередньо

згенерованому програмою числу), повідомте: «Неправильний

код, скриня вибухнула! Гру завершено.».

○ У разі правильного вводу повідомте: «Скарб ваш, ви

врятовані!».

У кінці гри (в блоці finally) завжди виводьте повідомлення:

«Гра завершена. Дякуємо за участь у пригоді!»

Код програми:

def pirate\_island\_escape():

print("Гра 'Втеча з острова піратів'")

secret\_code = random.randint(10, 99)

success = False

try:

# Етап 1: Перетин річки

print("\n=== Етап 1: Перетин річки ===")

wood = int(input("Скільки одиниць деревини використаєте для плоту (1-10)? "))

if wood < 3:

print("Деревини замало, пліт затонув!")

return

# Етап 2: Втеча від піратів

print("\n=== Етап 2: Втеча від піратів ===")

choice = input("Оберіть спосіб втечі (бігти/сховатися/битися): ").lower()

if choice not in ["бігти", "сховатися", "битися"]:

raise ValueError("Такого варіанту немає, пірати вас спіймали!")

# Етап 3: Відкриття скрині

print("\n=== Етап 3: Відкриття скрині зі скарбами ===")

code\_guess = int(input("Введіть двозначний код для відкриття скрині: "))

if code\_guess != secret\_code:

print("Неправильний код, скриня вибухнула! Гру завершено.")

else:

print("Скарб ваш, ви врятовані!")

success = True

except ValueError as e:

print(f"Помилка: {e}")

finally:

print("\nГра завершена. Дякуємо за участь у пригоді!")

if success:

print("Вітаємо з успішною втечею!")

pirate\_island\_escape()

Приклад роботи програми:

Гра 'Втеча з острова піратів'

=== Етап 1 ===

Скільки деревини? 5

=== Етап 2 ===

Оберіть спосіб: битися

=== Етап 3 ===

Введіть код: 42

Скарб ваш, ви врятовані!

Гра завершена. Вітаємо з успішною втечею!

Гра 'Втеча з острова піратів'

=== Етап 1 ===

Скільки деревини? 2

Деревини замало, пліт затонув!

Гра завершена. Дякуємо за участь!

Висновки

У лабораторній роботі 6 було розроблено серію ігор з покращеною логікою та обробкою помилок. Робота демонструє професійний підхід до обробки помилок і створення стабільних програм, які:

1. Не "падають" при некоректному вводі

2. Надають зрозумілі підказки користувачу

3. Забезпечують передбачувану поведінку в будь-яких сценаріях

4. Мають чітку завершальну фазу (навіть при помилках)