Міністерство освіти і науки України

Національний університет «Запорізька політехніка»

Кафедра програмних засобів

ЗВІТ

Дисципліна «Розробка прикладних програм»

Робота №1

Тема «Розроблення програмного забезпечення з графічним інтерфейсом на основі об’єктно-орієнтованого програмування мовою Python»

Виконав варіант 19

Студент КНТ-122 Онищенко О. А.

Прийняли

Викладач Дейнега Л. Ю.

2024

МЕТА РОБОТИ

Ознайомитися з принципами реалізації об’єктно-орієнтованого програмування у мові Python та навчитися використовувати його для розроблення програмного забезпечення.

Навчитися розробляти сучасні графічні інтерфейси користувача для програм, написаних мовою Python.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

Продаж квитків у кінотеатр з можливістю переглядати фільми, переглядати доступні та зайняті місця для перегляду заданого фільму у відповідній залі, бронювання та звільнення місць.

ТЕКСТИ ФАЙЛІВ

constants.py

```
# всі кольори обрано зі стандартних кольорів tkinter
GREEN_BUTTON="lawn green"
BLUE_BUTTON="dodger blue"
# ці кольори незавершені
# вони мають варіації з номером на кінці
COLORS=["AntiqueWhite","LightSteelBlue","khaki"]
```

movies.json

```
[
  {
    "name": "Страсті Христові",
    "description": "Рік 33 нашої ери. У римській провінції Юдея таємничий столяр на ім’я Ісус із Назарету починає сповіщати про прихід «Божого царства» і оточує себе групою покірних рибалок: Апостолів. Протягом століть єврейські люди чекали приходу Месії — провідної фігури, яка звільнить свою священну батьківщину та встановить новий порядок, заснований на справедливості. Вчення Ісуса привертає велику кількість послідовників, які визнають Його Месією. Стурбований ситуацією, синедріон за допомогою Юди Іскаріота, одного з дванадцяти
```

```
апостолів, арештовує Ісуса. Звинувачений у державній зраді проти Риму,
Христос передається Понтію Пілату, який, щоб уникнути бунту, засуджує
його на смерть на хресті як звичайного злочинця.",
    "year": 2004,
    "image":
"https://image.tmdb.org/t/p/w1280/v9f9MMrq2nGQrN7cHnQRmEq9lSE.jpg"
  },
  {
    "name": "Син Божий",
    "description": "Історія Ісуса Христа від Народження, до Його
вчення, Розп'яття і Вознесіння.",
    "year": 2014,
    "image":
"https://image.tmdb.org/t/p/w1280/1ICN5qakevvkAKI2MKWDhTFm9kh.jpg"
  },
  {
    "name": "Народження Христа",
    "description": "Фільм оповідає про життєвий шлях Діви Марії та
Йосипа, починаючи з моменту їх вигнання з Назарету та закінчуючи
прибуттям у Віфлеєм, де, як відомо, повинен народитися Ісус. У цій
непростій подорожі не раз перевірятиметься на міцність їх любов та
істинність переконань.",
    "year": 2006,
    "image":
"https://image.tmdb.org/t/p/w1280/vB55RLWmO2NYdGwL0XG30769cPO.jpg"
  }
]
```

objects.py

```
# головний об'єкт вікна програми
window=None
# глобально обраний фільм
movie=None
# глобально вибрана зала
room=None
# контейнери для фільмів, зал, місць
movies_container=None
rooms_container=None
seats_container=None
```

run.py

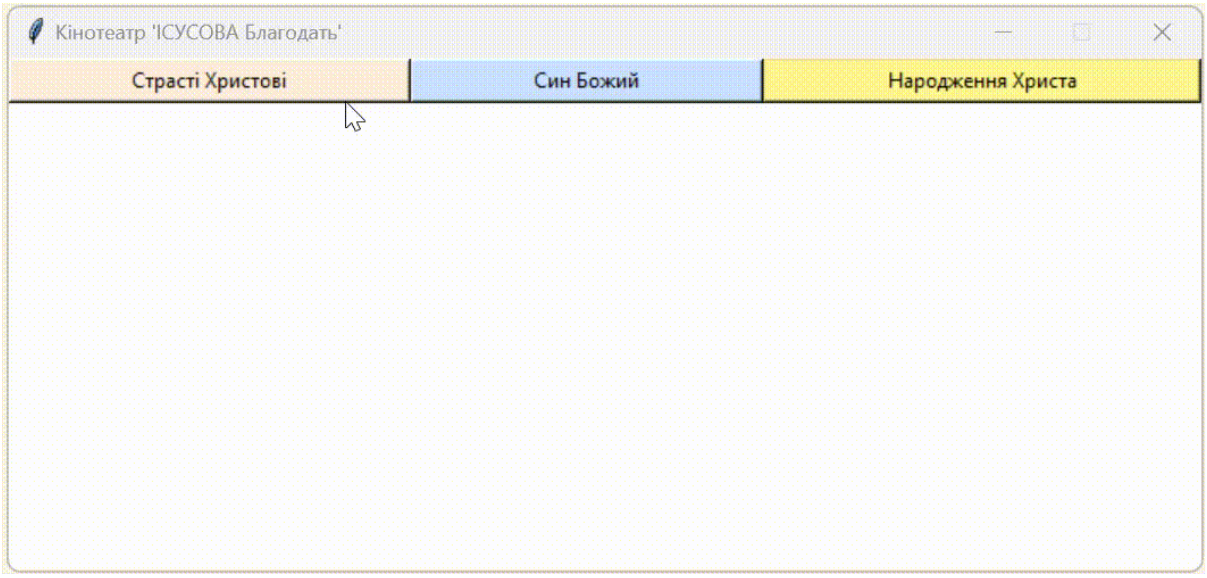
```
import os
import json
import random
from tkinter import *

from constants import *
from objects import *

class Movie:
```

```
    def __init__(self, name:str, description:str, year:int, image:
str):
        self.name:str=name
        self.description:str=description
        self.year:int=year
        # посилання на файл зображення
        self.image:str=image
        self.rooms=[
            # генеруємо три зали
            [
                # у кожній залі маємо три ряди сидінь
                [
                    # у кожному ряду маємо сім місць
                    # заповнюємо місця "випадковими" булевими
                    random.choice([0,1])
                    for seat in range(7)
                ]
                for row in range(3)
            ]
            for room in range(3)
        ]

class DataLoader:
    @staticmethod
    def load_movies() -> list[Movie]:
        """
        Завантажує дані про фільми з файлу JSON, конвертує їх до
об'єктів типу Movie

        > Список завантажених фільмів як масив об'єктів
        """

        current_folder:str=os.path.dirname(os.path.abspath(__file__))

movies_data_file_path:str=os.path.join(current_folder,"movies.json")
        movies:list[Movie]=[]

        with open(movies_data_file_path,mode="r",encoding="utf-8") as
f: movies_data:dict=json.load(f)
        for movie in movies_data:
            # оскільки ключі прочитаних об'єктів співпадають з полями
класу
            # їх можна одразу конвертувати у об'єкти
            movie_object:Movie=Movie(**movie)
            movies.append(movie_object)
        return movies

class Window(Tk):
    def __init__(self):
        super().__init__()
        self.geometry("700x300")
        self.resizable(0,0)
        self.title("Кінотеатр 'ІСУСОВА Благодать'")
        # при натисканні кнопки Esc програма закривається
        self.bind("<Escape>",lambda _: window.destroy())

class Container(Frame):
    def __init__(self,parent,expand:bool=0):
```

```
        super().__init__(parent,background="white")
        self.pack(fill=BOTH,expand=expand)

    def clear(self):
        for widget in self.winfo_children():
            widget.destroy()

def place_button(button_object:Button,command):
    button_object.config(command=command)
    button_object.pack(side=LEFT,expand=1,fill=BOTH)

def on_movie_select(movie_index:int):
    global movie
    movie=movies[movie_index]
    build_rooms()

def build_movies():
    movies_container.clear()
    for movie_index,movie in enumerate(movies):
        movie_button=Button(
            movies_container,
            # текст кнопки ставимо на назву фільму
            text=movie.name,
            # обираємо колір за номером фільму
            background=f"{COLORS[movie_index]}1"
        )
        place_button(movie_button,lambda
movie_index=movie_index:on_movie_select(movie_index))

def on_room_select(room_index:int):
    global room
    room=movie.rooms[room_index]
    build_seats()

def build_rooms():
    rooms_container.clear()
    for room_index,room in enumerate(movie.rooms):
        room_button=Button(
            rooms_container,
            # текст кнопки ставимо як номер зари
            text=f"Зала {room_index+1}",
            # колір ставимо як номер фільму
            background=f"{COLORS[movies.index(movie)]}1"
        )
        place_button(room_button,lambda
room_index=room_index:on_room_select(room_index))

def choose_button_color(value): return GREEN_BUTTON if not value else
BLUE_BUTTON

def on_seat_select(row_index,seat_index,button_object):
    global room
    # обираємо протилежне значення до поточного
    room[row_index][seat_index]=0 if room[row_index][seat_index] else 1
    # ставимо колір зелений якщо місце вільне, синій якщо зайняте

button_object.config(background=choose_button_color(room[row_index][sea
t_index]))
```

```
def build_seats():
    seats_container.clear()
    for row_index, row in enumerate(room):
        row_container=Container(seats_container,expand=1)
        for seat_index,seat in enumerate(room[row_index]):

seat_button=Button(row_container,background=choose_button_color(seat))
            place_button(seat_button,lambda
row_index=row_index,seat_index=seat_index,button_object=seat_button:on_
seat_select(row_index,seat_index,button_object))

window=Window()
movies=DataLoader.load_movies()

movies_container=Container(window)
rooms_container=Container(window)
seats_container=Container(window,expand=1)

build_movies()
window.mainloop()
```

РЕЗУЛЬТАТИ ВИКОНАННЯ



GIF Зображення 1.1 – Процес роботи з програмою

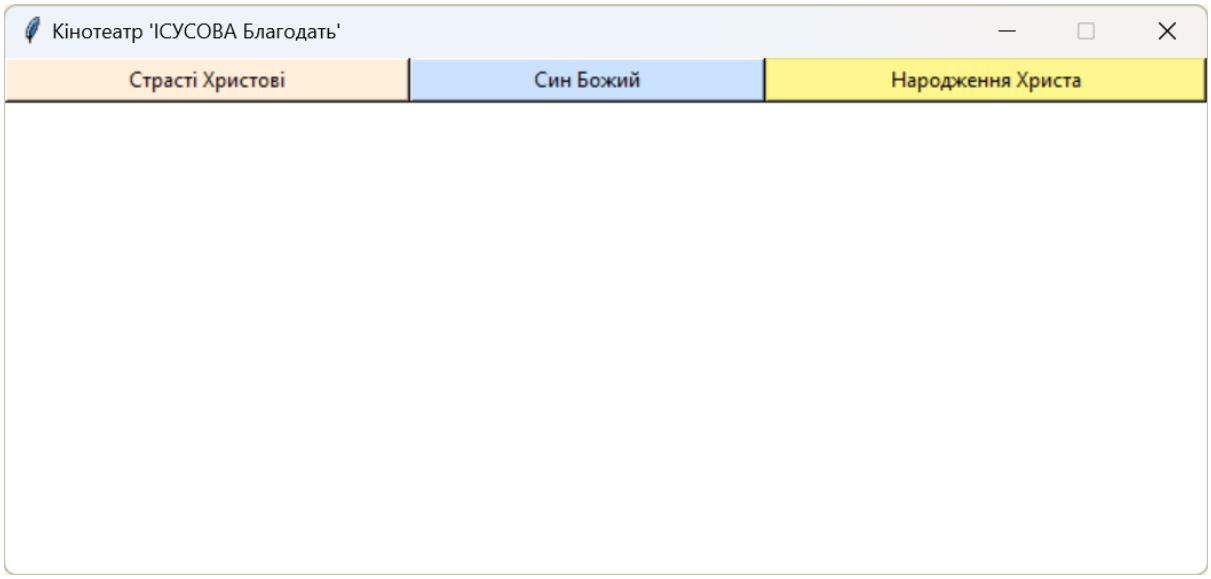


Рисунок 1.1 – Початковий вигляд програми

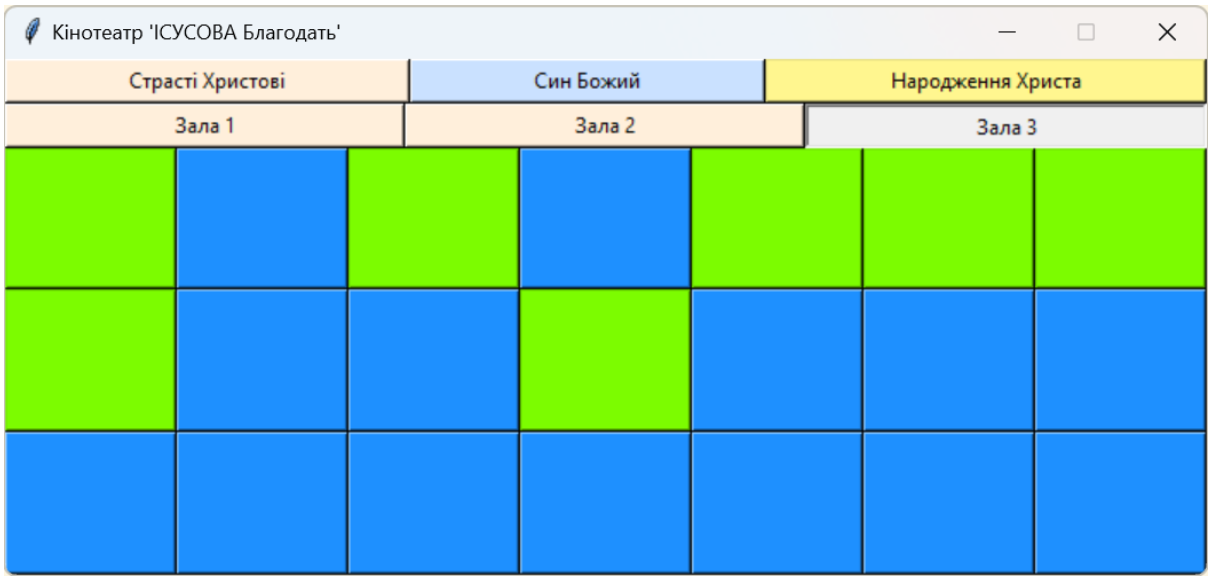


Рисунок 1.2 – Вигляд програми в користуванні

КОНТРОЛЬНІ ПИТАННЯ

Поняття модулю та пакету

Модуль це окремий файл з кодом.

Пакет це тека з файлом __init__.py і кількома модулями всередині.

Перетворення файлу у простір імен

При імпортуванні його до іншого файлу. Тоді до простору імен можна мати доступ через `назваФайлу.назваФункції`.

Назву простору імен можна змінити якщо додати до рядку імпорту параметр `as`. Наприклад: для імпортування модулю `math` з назвою `m` можна використати такий код: `import math as m`. Тоді для доступу до функції `sqrt` використовуємо наступний код: `m.sqrt(7)`

Пакети Python для побудови графічного інтерфейсу

- Tkinter
- PyQT
- Kivy

Засоби бібліотеки tkinter

Віджети:

- Button,
- Label,
- Entry,
- Checkbox,
- Combobox,
- Menu,
- Scale

Контейнери:

- Toplevel,
- Frame,
- Notebook

Розташувальники:

- pack(),
- grid(),
- place()

Повна документацію tkinter

Оголошення класу Python

Клас визначається ключовим словом `class`. Далі назва класу, краще у форматі `PascalCase` (кожне слово з великої літери, без розділення: `PlaneSeat`)

Спадкування: до оголошення класу додаємо дужки з назвою класу - або кількох - від яких успадковуємо: `class PlaneSeat(Seat)`

Конструктор: визначається оголошенням методу `__init__()`. Для кожного методу класу як перший параметр додається `self` для доступу до полів.

Статичний метод: додати декоратор `@staticmethod` перед оголошення методу.

Принципи об’єктно-орієнтованого програмування у Python

Статуси полів класу (private, public, protected) у пайтоні не передбачені, але можна визначити «приватне» поле додавши нижнє підкреслення перед назвою: `self._seatNumber`

Наслідування класів реалізується додаванням назв бажаних класів у дужки після назви класу при оголошенні: `class Plane(Transport)`

Поліморфізм реалізовується через використання зовнішньої функції. Приклад нижче.

Абстракцію реалізуємо використанням декоратора @abstractmethod. Приклад нижче.

Приклад поліморфізму

Код:

```
class Tvaryna:
    fraza = "ІСУС - ГОСПОДЬ"

    def kazhy(self):
        print("Звук ІСУСОВОЇ Дивовижної Благодаті")

class Golubka(Tvaryna):
    def kazhy(self):
        print(f"Голубка каже: {super().fraza}")

class Lev(Tvaryna):
    def kazhy(self):
        print(f"Лев каже: {super().fraza}")

def roby_zvuk(tvaryna: Tvaryna):
    tvaryna.kazhy()
```

Консоль:

```
Голубка каже: ІСУС - ГОСПОДЬ
Лев каже: ІСУС – ГОСПОДЬ
```

Приклад абстракції

Код:

```
from abc import abstractmethod

class Transport:
    def __init__(self,name:str):
        self.name=name

    @abstractmethod
    def show_kind(self): ...
```

```
class Plane(Transport):
    def __init__(self,name:str,kind:str):
        super().__init__(name)
        self.kind:str=kind

    def show_kind(self): print(f"Транспорт типу {self.kind}")

plane_object=Plane("Boeing 777","літак")
plane_object.show_kind()
```

Консоль:

```
Транспорт типу літак
```