МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет
«Запорізька політехніка»

МЕТОДИЧНІ ВКАЗІВКИ

до виконання лабораторних робіт
з дисципліни

“Системний аналіз”

для студентів спеціальностей
121 "Інженерія програмного забезпечення” та
122 “Комп’ютерні науки”
(всіх форм навчання)

2023

Методичні вказівки до виконання лабораторних робіт з дисципліни “Системний аналіз” для студентів спеціальностей 121 “Інженерія програмного забезпечення” та 122 “Комп’ютерні науки” (всіх форм навчання) / В.М. Льовкін. – Запоріжжя : НУ «Запорізька політехніка», 2023. – 69 с.

Автор: В.М. Льовкін, канд. техн. наук, доцент

Рецензент: А.О. Олійник, д-р. техн. наук, професор

Відповідальний
за випуск: С.О. Субботін, д-р. техн. наук, професор

Затверджено
на засіданні кафедри
програмних засобів

Протокол № 8
від “30” березня 2023 р.

ЗМІСТ

ВСТУП

Дане видання призначене для вивчення студентами всіх форм навчання основ системного аналізу та практичного засвоєння принципів побудови складних систем у процесі розроблення програмного забезпечення.

Відповідно до графіка студенти перед виконанням лабораторної роботи повинні ознайомитися з конспектом лекцій та рекомендованою літературою. Дані методичні вказівки містять тільки основні, базові теоретичні відомості, необхідні для виконання лабораторних робіт, тому для виконання лабораторної роботи та при підготовці до її захисту необхідно ознайомитись з конспектом лекцій та опрацювати весь необхідний матеріал, наведений в переліку рекомендованої літературі, використовуючи також статті у інтернет-виданнях та актуальних наукових журналах з інженерії програмного забезпечення.

Для одержання заліку з кожної роботи студент повинен у відповідності зі всіма наведеними вимогами розробити програмне забезпечення та оформити звіт, після чого продемонструвати на комп'ютері розроблене програмне забезпечення з виконанням всіх запропонованих викладачем тестів.

Усі завдання повинні виконуватись студентами індивідуально і не містити плагіату як в оформленому звіті так і в розробленому програмному забезпеченні.

Під час співбесіди при захисті лабораторної роботи студент повинен виявити знання щодо мети роботи, теоретичного матеріалу, методів виконання кожного етапу роботи, змісту основних розділів звіту з демонстрацією результатів на конкретних прикладах, практичних прийомів використання теоретичного матеріалу в розробленому програмному забезпеченні. Студент повинен вміти обґрунтувати всі прийняті ним проєктні рішення і рішення з аналізу і управління ризиками та правильно аналізувати і використовувати на практиці отримані результати. Для базової самоперевірки при підготовці до виконання і захисту роботи студент повинен відповісти на контрольні запитання, наведені наприкінці опису відповідної роботи.

1. ЛАБОРАТОРНА РОБОТА № 1 ЗАСТОСУВАННЯ СИСТЕМНОГО ПІДХОДУ ПІД ЧАС НАПИСАННЯ ПРОГРАМНОГО КОДУ

1.1. Мета роботи

1.1.1 Ознайомитись з основними можливостями, парадигмами, типами даних, синтаксичними особливостями та принципами мови програмування Python.

1.1.2 Навчитися розробляти програми процедурного, об'єктно-орієнтованого програмування на основі системного підходу.

1.2. Короткі теоретичні відомості

1.2.1 Засоби розроблення програм мовою Python

Python – популярна високорівнева мова програмування, яка використовується для розроблення програм та створення прикладних сценаріїв у різноманітних галузях.

Написання програмного коду на Python включає не тільки мову програмування, але й інтерпретатор, який необхідно встановити для початку роботи.

На даний момент існують наступні основні реалізації мови програмування Python:

– CPython – стандартна реалізація мови програмування Python, написана мовою C;

– Jython – альтернативна реалізація мови Python, побудована з Java-класів, які виконують компіляцію програмного коду мовою Python у байт-код Java, дозволяючи створювати графічний інтерфейс з використанням механізмів Java тощо;

– IronPython – реалізація, призначена для забезпечення інтеграції програм, написаних мовою Python, з додатками, створеними для роботи в середовищі Microsoft .NET Framework операційної системи Windows, його відкритій альтернативі Mono, таким чином дозволяючи програмам мовою Python грати роль як клієнтських, так і серверних компонентів, доступних для інших мов програмування .NET;

– PyPy – реалізація інтерпретатора, написана за допомогою мови Python.

Дані реалізації дозволяють виконувати інструкції в інтерактивному режимі або створювати окремі модулі.

Останні версії операційних систем сімейства Linux (Debian, Ubuntu, Fedora) мають встановлений компонент Python за замовчанням.

Таким чином, процес інсталяції інтерпретатора залежить від операційної системи та обраної реалізації. Стандартним інтерпретатором є CPython, який зазвичай використовується за замовчуванням в операційних системах сімейства Linux. Завантажити його можна з сайту http://www.python.org.

Окрім використання стандартного інтегрованого середовища розробки IDLE з пакету CPython, яке дозволяє редагувати, запускати, зневаджувати програми мовою Python, можна також розробляти програми мовою Python за допомогою сучасних IDE. Розроблення програм мовою Python підтримується зокрема наступними IDE:

– Eclipse;
– PyCharm;
– Komodo;
– NetBeans;
– PythonWin;
– Wing;
– Microsoft Visual Studio.

Для створення програмних проєктів на Python у середовищі Eclipse необхідно встановити плагін PyDev за допомогою меню Help → Eclipse Marketplace… або завантажити архів з плагіном та помістити дані в теку eclipse\dropins.

Після того, як плагін встановлено, можна створити PyDev Project (рис. 1.1), звідки можна задати налаштування інтерпретатора, який буде використовуватись для роботи з даним проєктом.

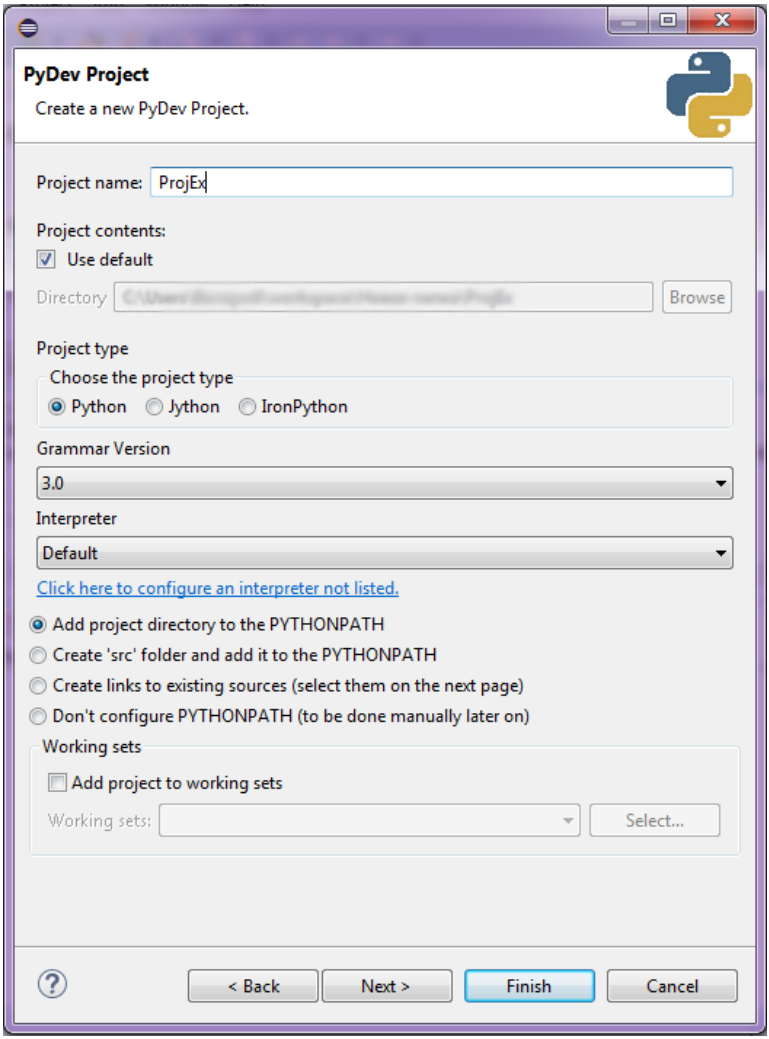

Рисунок 1.1 – Створення нового проєкту PyDev

У створений проєкт PyDev можна додати пакет, модуль PyDev або окремий файл (рис. 1.2).

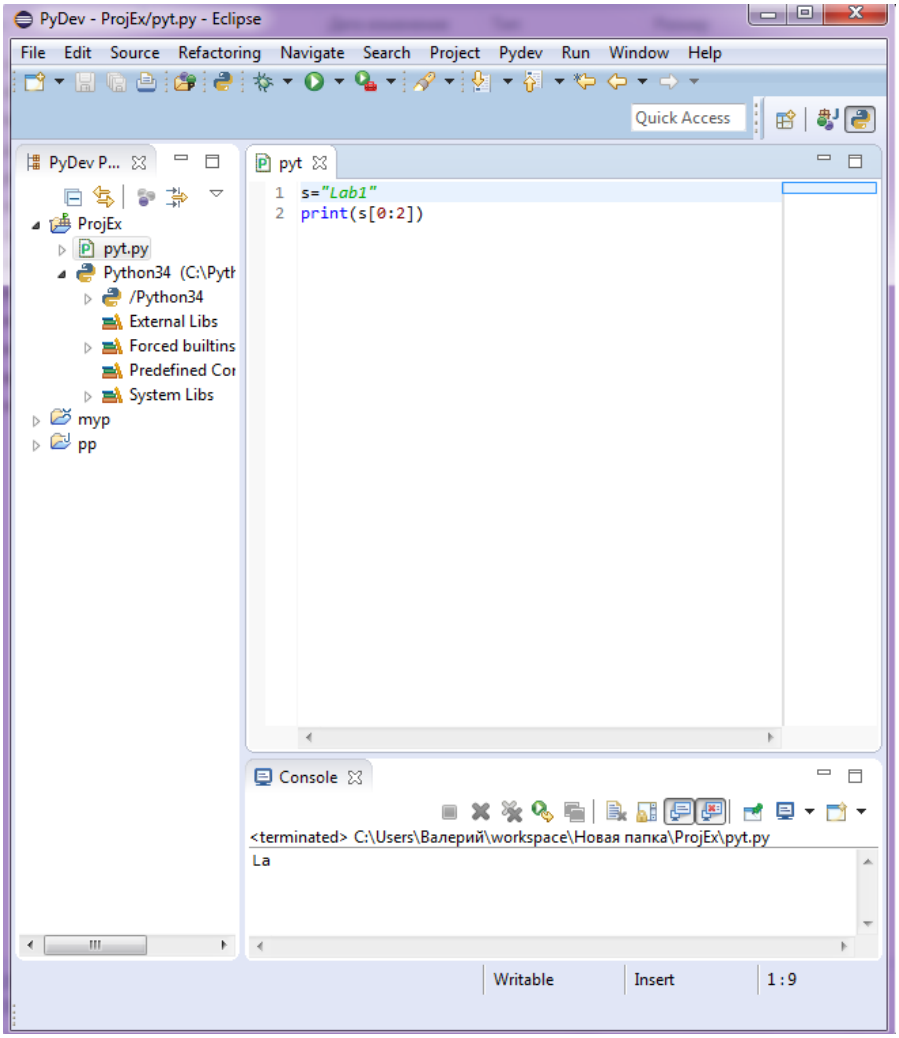

Рисунок 1.2 – Робота над проєктом PyDev мовою програмування Python у IDE Eclipse

1.2.2 Основи мови програмування Python

Python має потужну довідкову систему. Довідкові дані можна отримати як за допомогою документації Python, доступної зокрема з

меню IDLE, так і за допомогою відповідних функцій через інтерпретатор.

Функція dir повертає перелік всіх доступних атрибутів заданого об’єкту:

– для того щоб дізнатись про всі інструменти, доступні в деякому бібліотечному модулі, необхідно імпортувати його та передати ім’я даного модуля функції dir;

– для того щоб дізнатись про всі атрибути об’єктів вбудованих типів, необхідно передати функції dir літерал або існуючий об’єкт відповідного типу.

Для того щоб отримати інформацію про призначення того чи іншого методу, необхідно передати його ім’я функції help.

У мові Python використовується динамічна типізація: типи даних визначаються автоматично, змінну не потрібно оголошувати.

У таблиці 1.1 представлені основні вбудовані типи об’єктів мови програмування Python.

Таблиця 1.1 – Основні вбудовані об’єкти в мові програмування Python

Тип	**Приклад**
Число	15, -103, 14.28
Рядок	'example', s2="it's"
Перелік	[1, "abc"]
Словник	{'rec1': 10, 'rec2': 22}
Кортеж	(-23, 15, "text")
Множина	set(‘str’), {‘s’, ‘t’, ‘r’}
Додаткові типи	Функції, файли, модулі, класи

Таблиця 1.1 містить далеко не повний перелік об’єктів, тому що об’єктами є всі дані, оброблення яких виконується в програмах мовою Python.

Приклад програми з використанням об’єктів числових та рядкових типів:

Python

```
percentage = 20.5 / 100
sum_before_taxes = 7052.30
res = sum_before_taxes * percentage
mess = 'calculated:'
print(mess+str(res))
```

У таблиці 1.2 представлені шаблони виклику деяких вбудованих рядкових методів.

Таблиця 1.2 – Основні рядкові методи

Метод	**Опис**
str.capitalize()	Обробляє рядок, перетворюючи його першу літеру на велику, а всі інші залишаючи маленькими
str.count(value, start, end)	Виконує обчислення кількості входжень тексту (підрядка) в рядку
str.endswith(value, start, end)	Виконує перевірку завершення даного рядка заданим набором символів
str.find(value, start, end)	Виконує пошук тексту (підрядка) в рядку зліва
str.format(value1, value2...)	Форматування рядка
str.isalnum()	Перевірка того, чи складається рядок повністю з літер або цифр
str.isalpha()	Перевірка того, чи складається рядок повністю з літер
str.islower()	Перевірка того, чи складається рядок повністю з символів у нижньому регістрі
str.isnumeric()	Перевірка того, чи складається рядок повністю з цифр

Продовження таблиці 1.2

Метод	**Опис**
str.isprintable()	Перевірка того, чи складається рядок повністю з друкованих знаків
str.isspace()	Перевірка того, чи складається рядок повністю з пробільних символів
str.isupper()	Перевірка того, чи складається рядок повністю з символів у верхньому регістрі
str.lstrip(characters)	Обробляє рядок, вилучаючи з нього тільки початкові, розташовані поспіль символи, які є одними з символів з рядка characters
str.partition(sep)	Знаходить входження підрядку в рядок і розділяє за цим входженням рядок на три частини
str.replace(oldvalue, newvalue, count)	Виконує заміну входжень підрядків oldvalue в рядку на рядок newvalue, повторюючи це count разів
str.rfind(value, start, end)	Виконує пошук тексту (підрядка) в рядку, починаючи з кінця
str.rpartition(sep)	Знаходить останнє входження підрядку в рядок і розділяє за цим входженням рядок на три частини
str.rstrip(characters)	Обробляє рядок, вилучаючи з нього тільки кінцеві, розташовані поспіль символи, які є одними з символів з рядка characters
str.split(separator, maxsplit)	Виділяє слова з рядку, розділяючи їх за допомогою символів з рядка separator, у кількості maxsplit
str.splitlines(add)	Розділяє текст на рядки, залишаючи символи відділення або ні (визначає параметр add)

Продовження таблиці 1.2

Метод	**Опис**
str.startswith(value, start, end)	Виконує перевірку початку даного рядка заданим набором символів
str.strip(characters)	Обробляє рядок, вилучаючи з нього початкові, розташовані поспіль символи, які є одними з символів з рядка characters, та аналогічні символи, розташовані в кінці рядка
str.swapcase()	Обробляє рядок, змінюючи регістр всіх символів (верхній на нижній, нижній – на верхній)
str.upper()	Обробляє рядок, збільшуючи регістр всіх символів

У таблиці 1.3 представлені основні операції, які використовуються під час роботи з переліками.

Таблиця 1.3 – Літерали переліків та операції

Операція	**Опис**
l = [20, 7, 1004, 2]	Створення переліку
l = list('text')	Створення переліку за допомогою конструктора
len(l)	Отримання довжини переліку
l[2]	Витягання значення з переліку за індексом
l[1:3]	Отримання зрізу
l+l2	Конкатенація переліків
l.append(52)	Внесення елемента в перелік
l.count(2)	Визначення кількості входжень елемента в перелік
l.extend([9, 10])	Розширення переліку

Продовження таблиці 1.3

Операція	**Опис**
l.index(20)	Визначення індексу входження елемента в перелік
l.insert(1, 12)	Внесення елемента в перелік на задану позицію (перший аргумент)
l.pop()	Вилучення останнього елемента з переліку
l.remove(1)	Вилучення елемента з переліку за значенням
l.reverse()	Зміна порядку елементів у переліку на протилежний
l.sort()	Сортування переліку
del L[1]	Вилучення елемента з переліку за позицією

У таблиці 1.4 представлені основні операції, які використовуються під час роботи зі словниками.

Таблиця 1.4 – Літерали словників та операції

Операція	**Опис**
v = {'attr1': 10, 'attr2': 20}	Створення словника з парами ключ-значення
v = dict(attr1 = 'Name', attr2=100)	Створення словника за допомогою конструктора
v['attr1']	Витягання значення зі словника за ключем
'attr1' in v	Перевірка існування ключа в словнику
v.copy()	Копіювання словника
v.items()	Отримання переліку пар з ключів і значень зі словника

Продовження таблиці 1.4

Операція	**Опис**
v.keys()	Отримання переліку ключів зі словника
v.values()	Отримання переліку значень зі словника
v.update(v2)	Об’єднання словників
len(v)	Отримання довжини словника
v[key] = 32	Внесення значення в словник за ключем
del v[key]	Вилучення ключа зі словника

У таблиці 1.5 представлені основні операції, які використовуються під час роботи з кортежами.

Таблиця 1.5 – Літерали кортежів та операції

Операція	**Опис**
t = ('value', 18.3, 19)	Створення кортежу
t = tuple('str')	Створення кортежу за допомогою конструктора
len(t)	Отримання довжини кортежу
t[2]	Витягання значення з кортежу за індексом
t[1:3]	Отримання зрізу
t+t2	Конкатенація кортежів
t.count(19)	Визначення кількості входжень елементу в кортеж
t.index(19)	Визначення індексу входження елементу в кортеж

У таблиці 1.6 представлені основні операції, які використовуються під час роботи з файлами.

Таблиця 1.6 – Основні операції над файлами

Операція	**Опис**
fstream = open('data.txt', 'r')	Відкриття файлу для його читання (rb для читання двійкових даних)
data = fstream.read()	Прочитання всього файлу одразу
data = fstream.read(N)	Прочитання N символів (байт) файлу
data = fstream.readline()	Прочитання текстового рядка
ofstream.write(data)	Запис рядка у файл
ofstream.close()	Закриття файлу
ofstream.seek(N)	Зміщення поточної позиції в файлі

У таблиці 1.7 представлені основні інструкції, які використовуються в процесі створення програм мовою Python.

Таблиця 1.7 – Інструкції мови Python

Інструкція	**Приклад**
Присвоювання	a, b = 'text', 10
Виклик функції	fun(data)

break	while True:
	  if not fun():
	  break

continue	while True:
	  if fun():
	  continue
	  print(message)

def	def fun(a, *b): return a+b[0]
if (elif-else)	if 't' in info: print(message)
import	import lib

Продовження таблиці 1.7

Інструкція	**Приклад**
for	for x in onelist: print(x)
from	from lib import cl

global	x = 15
	def fun():
	  global x
	  x = 'value'

pass	def fun(): pass

while	while l:
	  print(message)
	  l.pop()

yield	def fun(n):
	  for i in n:
	  yield i*2

1.3. Завдання на лабораторну роботу

1.3.1. Ознайомитись з теоретичними відомостями, необхідними для виконання роботи.

1.3.2. Розробити програмне забезпечення у відповідності з індивідуальним завданням, узгодженим з викладачем. Завдання б) обов'язково передбачає створення відповідних **класів** для роботи з предметною областю тільки через них.

1.3.2.1 Завдання:

а) Файл містить масив рядків, кожний з яких представляє собою запис про обліковий запис користувача, а кожне поле відділяється від іншого двокрапкою. Відомо, що перше поле зберігає логін, п'яте – прізвище та ім'я користувача, а шосте – пароль. Запросити логін користувача. Якщо логін невідомий, то попросити ввести логін заново. Передбачити вихід з програми, якщо буде введено слово

“bye”. Якщо логін знайдено в масиві, то вивести вітання з зазначенням ім’я (перше слово з відділених комами) та попросити ввести пароль. За результатами перевірки пароля вивести повідомлення про успішну авторизацію, або надати користувачу додаткову спробу.

б) Надання доступу до інформації про власників нерухомості в місті Запоріжжя. Має надаватися можливість внесення відповідних даних, редагування та виконання пошуку. Для кожного власника має розраховуватися податок, який він має сплатити за рік. Податок залежить від загальної площі всіх об’єктів одного власника.

1.3.2.2 Завдання:

а) Дані про результати роботи підприємств формують файл, кожний запис якого представлений наступними параметрами: ідентифікаційний код суб’єкта господарської діяльності, назва підприємства, директор, адреса, розмір доходів за рік, розмір витрат за рік. Організувати масив, кожний елемент якого містить відповідний набір параметрів. Вивести у другий файл перелік підприємств за зменшенням сплаченого податку на прибуток. У випадку неоднозначності – за отриманими доходами. Податок на прибуток розраховується в розмірі 18 % від отриманого підприємством прибутку.

б) Контроль за школярами. Кожний школяр (як і всі користувачі) ідентифікується системою, результати його навчання (присутність, оцінки) визначаються вчителями. Батьки можуть переглядати інформацію тільки про власних дітей.

1.3.2.3 Завдання:

а) Деяка, задана користувачем, кількість монет лежить перед двома гравцями. Кожен з них може взяти за один раз кількість монет, задану елементами множини (наприклад, 1, 3, 5). Гравці виконують кроки по черзі. Визначити, який гравець (перший чи другий) виграє, якщо буде виконувати для цього логічні кроки.

б) Підтримка процесу тестування. Адміністратор може створювати нові тестування, додавати запитання до визначеного тестування. Звичайні користувачі можуть проходити тестування за вибраним тестом. Кожне тестування відбувається шляхом випадкового вибору набору запитань зі всіх доступних (кількість запитань визначається в параметрах тестування).

1.3.2.4 Завдання:

а) Матриця, кожний елемент якої є рівнем інтенсивності пікселя деякого зображення, представлена у файлі. Передбачити функції, які дозволять обробити дані з файлу за допомогою фільтру заданого користувачем розміру: розмір поля для кожного пікселя може складати 3 на 3, 5 на 5, 7 на 7 пікселів тощо. Медіаною набору елементів {a, b, c}, відсортованого за збільшенням значень (a < b < c), є значення b, а набору {a, b, c, d} – значення (b + c)/2. Значення елементу mas[i][j] змінюється на медіану з набору значень {mas[i-1][j--1], mas[i-1][j], mas[i-1][j+1], mas[i][j-1], mas[i][j], mas[i][j+1], mas[i+1][j-1], mas[i+1][j], mas[i+1][j+1]}.

б) Облік надання послуг користування Інтернетом. Представники інтернет-провайдера можуть визначати наявні тарифи. Абоненти можуть обирати необхідний тариф. Абоненти можуть переглядати власну історію вибору тарифів (детальну та агреговану). Обчислюється сума до сплати для кожного користувача за кожний місяць.

1.3.2.5 Завдання:

а) Відомо, що у заданому тексті можуть зустрічатися дужки. З заданого рядка вилучити текст, що розташований в дужках (разом з дужками та відповідною кількістю пробілів). Передбачити можливість ситуації, коли дужки закриті невірно. У випадку такої ситуації вилучення тексту, розташованого після відкритої дужки або перед закритою дужкою, не виконувати. Повідомити, якщо хоча б одна дужка не закрита або не відкрита вчасно.

б) Доставка вантажів. Користувачам надається можливість оформлення замовлень на доставку товарів, відстеження поточного стану замовлення. Оператори можуть змінювати стан замовлення. Для кожного замовлення обирається відповідний автомобіль для доставки.

1.3.2.6 Завдання:

а) У файлі міститься повідомлення. В іншому файлі міститься набір чисел, які задають позиції. Виконати шифрування повідомлення, розбиваючи його на блоки та розташовуючи символи в кожному такому блоці в порядку, заданому другим файлом.

б) Керування бібліотекою. Повинна надаватися можливість додавати, вилучати та шукати книги та читачів. Читач може брати не більше п’яти книжок. Кожна книга міститься в певній кількості

екземплярів, наявність вільних має перевірятися під час видачі. У разі використання книги довше заданого періоду часу нараховується пеня.

1.3.2.7 Завдання:

а) У файлі міститься арифметичний вираз, що може включати цілі числа, арифметичні оператори (+, –, *), дужки. Обчислити значення даного виразу та дописати його в файл у вигляді «=x».

б) Оренда автомобілів. Адміністратори визначають перелік наявних автомобілів, їх вартість тощо. Орендар може виконувати пошук за необхідними параметрами, брати авто в оренду. Вартість послуги розраховується автоматизовано перед початком оренди та перераховується після повернення автомобіля.

1.3.2.8 Завдання:

а) Робот наносить постріли по бочках, що розташовані перед ним. Розташування кожної з них моделюється прямокутною формою і задається відповідними координатами (лівого верхнього та правого нижнього кута) в тривимірному просторі. Відповідні дані зберігаються в файлі. Яку кількість варіантів має робот для першого пострілу? Вважати, що робот може поцілити в бочку, якщо вона повністю не перекрита іншою бочкою.

б) Відбір студентів для участі в освітніх проєктах в університеті. Інформація про кандидатів включає зокрема його бал сертифікату зі знання англійської мови, середній бал навчання та спеціальність конкурсу. Адміністратор вносить в систему інформацію про кількість місць за кожною спеціальністю.

1.3.2.9 Завдання:

а) У файлі міститься інформація про телефони абонентів, кожний запис складається з прізвища абонента, дати підключення, номеру телефону. Визначити функції, які дозволять виводити інформацію про номер телефону за заданим прізвищем абонента, визначати кількість підключених телефонів з заданого користувачем року, а також зчитувати дані з файлу та заповнювати файл.

б) Замовлення таксі. Клієнту надається можливість обрати автомобіль або водія, отримати інформацію про орієнтовну вартість за даними замовлення (маршрут). Кожне замовлення зберігається разом з фактичною вартістю. Замовлення можна тільки редагувати. Водіїв та автомобілі можна вносити в систему.

1.3.2.10 Завдання:

а) У кожному рядку файлу містяться багаточлени. Кожен одночлен у багаточлені відділяється від іншого одночлена арифметичною операцією (додавання або віднімання). Кожен одночлен містить набір змінних, імена кожної з яких складаються з однієї літери, та чисел. Перше число в одночлені – коефіцієнт, який може бути пропущено. Наступні числа, що є цілими числами (необов'язково невід'ємними), – ступені змінних. Виконати приведення багаточленів.

б) Оформлення патентів на винаходи. Надається можливість виконувати пошук зареєстрованих патентів за критеріями. Звичайні користувачі можуть подавати заявку на реєстрацію патенту, ідентифікуючи його відповідним чином, визначаючи відповідального патентного повіреного, переглядати статус заявки та редагувати її. Патентний повірений може переглядати надіслані йому заявки, редагувати їх та реєструвати в системі в якості патентів.

1.3.2.11 Завдання:

а) У файлі міститься набір цілих чисел у десятковій системі числення. Виконати шифрування даних, виконуючи перетворення чисел у двійкову систему та заміну позицій бітів кожного числа. Спосіб заміни бітів задається другим файлом, що містить пари чисел, які позначають позиції бітів, які необхідно замінити між собою. У результаті сформувати третій файл. Аналогічно виконати дешифрування.

б) Підтримка навчання студентів, які можуть підписуватися на проходження курсів. У межах кожного курсу студенти можуть переглядати лекції, отримувати і виконувати завдання. Кожен курс та окремі його лекції доступні протягом заданого викладачем проміжку часу.

1.3.2.12 Завдання:

а) Розробити програму, яка виділяє всі слова, що зустрічаються в файлі, обчислює кількість входжень кожного слова. Регістр при цьому не враховувати. Відсортувати перелік слів з файлу в алфавітному порядку та вивести даний перелік на екран з зазначенням кількості входження кожного слова.

б) Керування рахунками в банку. Менеджер може керувати (переглядати, редагувати, зберігати) рахунками, послідовно переміщуючись між ними за PIN-кодом (а також на перший та

останній код з можливістю створення нового) або виконуючи пошук за номером телефону, а клієнт може керувати тільки власним рахунком та змінювати пароль.

1.3.2.13 Завдання:

а) Користувач задає натуральне число, яке визначає довжину рядка. Вирівняти кожний рядок у файлі, тобто додати у відповідний рядок рівномірно стільки пробілів між словами, щоб довжина рядка стала рівною заданому натуральному числу. Перед вирівнюванням спочатку вилучити зайві пробіли на початку речення та між словами. Вивести вирівняні рядки з першого файлу в другий файл.

б) Футбольний тоталізатор. Визначаються футбольні матчі та варіанти результатів. Користувачі роблять ставки, визначаючи результат та суму грошей. Результати визначаються випадковим чином після настання відповідної дати. За отриманими результатами визначається виграш кожного учасника.

1.3.2.14 Завдання:

а) Виконати дії над двомірним масивом *N*x*N*. Параметр *N* задається користувачем з клавіатури. Заповнити даний масив послідовно даними за спіраллю, починаючи з лівого верхнього кута та рухаючись за годинниковою стрілкою:

1 2 3 4
12 13 14 5
11 16 15 6
10 9 8 7

В якості даних обрати цілі числа з інтервалу [*a*; *b*], де *a* і *b* задаються користувачем.

б) Підтримка спільних поїздок автомобілем. Власники автомобілів визначають маршрут поїздки, дати, кількість наявних місць, вартість тощо, підтверджують бронювання. Звичайні користувачі можуть виконувати пошук та бронювати місця.

1.3.2.15 Завдання:

а) У числовому ребусі зашифровано цифри літерами. Ребус визначає операцію складання двох чисел та її результат. При цьому однакові цифри зашифровано однаковими літерами. Заданий ребус міститься в файлі. Розв’язати ребус та результат записати в той самий файл додатковим рядком.

б) Порівняння цін на ліки. Користувачі можуть виконувати пошук наявності ліків у різних аптеках шляхом введення назви ліків та необхідної кількості. За заданими параметрами виконується пошук з сортуванням за збільшенням вартості ліків.

1.3.2.16 Завдання:

а) Виділити з заданого тексту всі електронні адреси і зберегти їх, запитуючи у користувача ім'я власника для кожної скриньки, якщо дана інформація не зберігається в окремому файлі. Вхідні та вихідні дані мають зберігатися у файлі. Надати можливість користувачу надіслати електронний лист, зазначаючи ім'я адресата та вводячи тему і текст листа. Сформований лист виводити у файл.

б) Підтримка процесу винаймання робітників. Менеджери визначають посади, на які необхідно винайняти працівників, вимоги до них, переглядають резюме кандидатів на кожну посаду та оцінюють їх. Звичайні користувачі можуть подавати власні резюме та переглядати результати розгляду резюме.

1.3.2.17 Завдання:

а) Виконати аналіз XML-файлу. Виділити з файлу всі користувацькі теги та сформувати з них словник. Враховувати, що теги можуть містити атрибути. Для кожного тегу підрахувати кількість його входжень у файл. Перевірити правильність використання тегів: кожен відкритий тег має бути відповідним чином закритий. Вивести повідомлення про результати аналізу, що містить помилку, якщо її було знайдено.

б) Підтримка роботи туристичного агентства. Клієнти можуть переглядати доступні пропозиції та оформляти замовлення з врахуванням можливої знижки (в залежності від історії замовлень). Менеджери можуть переглядати замовлення, змінювати їх статус та переглядати статистику.

1.3.2.18 Завдання:

а) Сформувати тлумачний словник, дозволяючи користувачу додавати нові слова в словник, змінювати існуючі (як слова, так і тлумачення) та виконувати пошук слів у словнику. Надавати користувачу можливість вводити одразу декілька слів (словосполучення), пропонуючи йому тлумачити даний набір слів як словосполучення або тлумачити окремі слова. Визначити всі випадки, коли в тлумаченні слова використовується дане слово.

б) Менеджер фінансів. Користувачі можуть керувати власними фінансами шляхом додавання записів про доходи та видатки за групами, переглядати послідовно власні записи, переглядати найбільші витрати та доходи.

1.3.2.19 Завдання:

а) Файл містить перелік повних адрес файлів (ім'я диску, список каталогів, ім'я файлу та розширення). Виділити з кожної адреси ім'я файлу, розширення та адресу першого каталогу. Перевірити для кожного файлу, чи існує він. Вивести у файл, ім'я якого формується з імені початкового файлу додаванням постфіксу «srt», перелік файлів, які існують на диску, згрупувавши їх за форматами та відсортувавши в алфавітному порядку за іменем файлів. Імена файлів виводити у форматі «/<перший каталог>/…/<ім'я файлу>», сортуючи за розширенням та шляхом.

б) Продаж квитків у кінотеатр з можливістю переглядати сеанси, переглядати доступні та зайняті місця для перегляду заданого сеансу у відповідному залі, бронювання та звільнення місць. Інформація про нові сеанси може додаватися.

1.3.2.20 Завдання:

а) У файлі містяться розміри ящиків (ширина, висота, глибина), які є в наявності. Сформувати стос ящиків таким чином, щоб кожен нижній ящик був за висотою, шириною та глибиною більше ящика, розташованого на ньому. Вважати висоту стосу рівною сумі висот ящиків. Сформувати стос з найбільшою висотою. Ящики можна повертати.

б) Оформлення послуг. Користувачі можуть визначати власні послуги, період їх доступності, вартість, географічну зону їх доступності (за поштовим індексом). Замовники послуг можуть виконувати пошук, задаючи власне розташування, оформлювати замовлення послуг.

1.3.2.21 Завдання:

а) У файлі міститься словник. Користувач задає два слова. Визначити найкоротшу послідовність перетворень, яка дозволить перетворити перше задане користувачем слово на друге. Кожне перетворення повинно змінювати тільки одну літеру в слові і приводити до отримання іншого коректного слова зі словника. Якщо

таку послідовність знайти неможливо, вивести відповідне повідомлення.

б) Підведення підсумків виборів. Кожна політична партія характеризується певною інформацією, яка може редагуватися. Кожна дільниця характеризується номером, адресою, головою та членами. За кожною дільницею вносяться результати: кількість голосів, віддана за кожну партію. Надається можливість підведення підсумків шляхом визначення партій, які набрали більше 5 % від загальної кількості відданих голосів, та визначення набраних ними відсотків.

1.3.2.22 Завдання:

а) У файлі з розширенням *.cpp міститься текст програми. Виділити всі функції, які зустрічаються в даній програмі. Для заданого користувачем імені функції навести її тіло. Для кожної функції окремо навести всі змінні, які використовуються в даній функції. Окремо виділити всі глобальні змінні та константи з даної програми.

б) Керування комунальними послугами. Кожен рахунок має номер, власника, адресу. Рахунки можуть додаватися та редагуватися. За кожним рахунком вноситься інформація за кожен місяць щодо показань лічильника та автоматично розраховується сума до сплати. Окремо вноситься інформація про сплачені кошти. У випадку наявності заборгованості розраховується пеня. Надається можливість перегляду інформації про історію сплати, поточний стан рахунку.

1.3.2.23 Завдання:

а) У файлі міститься деякий текст, редагування якого необхідно виконати. У другому файлі містяться слова та словосполучення, які необхідно вилучити з даного тексту. Деякі слова або слова у словосполученнях можуть задаватися маскою (‘*’ позначає будь-яку кількість будь-яких символів, ‘?’ позначає будь-який символ). Програма повинна працювати у двох режимах: вилучаються тільки відповідні слова і словосполучення або вилучаються цілі речення, у яких дані слова або словосполучення зустрічаються.

б) Керування посилками. Кожна посилка прив’язана до певного користувача та відзначається станом. Стан змінюється операторами. Користувачі можуть переглядати власні посилки за заданим станом та тільки після завершення доставки виставляти оцінки. Надається можливість підрахунку середнього балу.

1.3.2.24 Завдання:

а) Сформувати всі *k*-елементні підмножини множини, члени якої задані у вигляді слів. Слова містяться у файлі. Вивести всі сформовані підмножини для заданого користувачем *k*. Якщо *k* більше кількості слів, вивести відповідне повідомлення.

б) Користувачі можуть виконувати пошук наявності товарів у різних магазинах шляхом введення назви товару та/або необхідних їх параметрів (вартість, категорія тощо). За заданими параметрами виконується пошук з сортуванням за збільшенням або зменшенням вартості, за алфавітним представленням назв магазинів. Товари за кожним магазином можуть додаватися, а їх вартість може змінюватися

1.3.2.25 Завдання:

а) Знайти всі варіанти розміщення восьми ферзів на шаховій дошці таким чином, щоб ніякі дві фігури не розміщувались на одній діагоналі, горизонталі або вертикалі. Враховувати не тільки головні діагоналі. Всі варіанти вивести на екран послідовно з відповідним схематичним позначенням.

б) Винаймання велосипедів. Кожен велосипед характеризується моделлю, станом (на станції або в дорозі). Інформація про нові велосипеди може вноситися. Під час винаймання користувач обирає велосипед зі всіх наявних на станції. Під час повернення велосипед фіксується за станцією повернення та розраховується сума до сплати (за часом ви наймання). Сума на рахунку користувача може змінюватися та списується автоматично після повернення.

1.3.2.26 Завдання:

а) У файлі міститься інформація про кількість, координати вікон, які відкрито в операційній системі, та їх розташування за порядком (1 – знизу). Координати кожного вікна задаються абсцисою і ординатою відносно лівого верхнього кута та шириною і висотою вікна. Інформація про вікна задана в файлі в довільному порядку, тобто порядок їх розташування визначає тільки відповідний показник, заданий для кожного вікна у файлі. Визначити, які з вікон видно користувачу на екрані. Вважати, що вікно видно, якщо видно хоча б один з його пікселів.

б) Визначення абітурієнтів, рекомендованих до зарахування в університет. Абітурієнти визначаються прізвищем, іменем, по

батькові, телефоном, балами сертифікату ЗНО, балом навчання, додатковими балами (тільки останні три параметри можуть редагуватися після реєстрації), спеціальностями, на які він подає заявки. Спеціальності визначаються початково разом з кількістю наявних місць. Рекомендовані до зарахування визначаються за загальним балом за кожною спеціальністю. Має виконуватися перевірка введення некоректних результатів.

1.3.2.27 Завдання:

а) У файлі міститься набір слів, по одному в кожному рядку. Визначити пару слів, що мають найдовшу послідовність літер з кінця, однакових у даних слів. Обчислити довжину такої послідовності. Вивести отримані результати в файл. Знайти відповідну пару з вхідного файлу для заданого користувачем слова.

б) Продаж квитків на концерти. Кожен концерт характеризується певною кількістю квитків кожної категорії. Користувачі можуть купляти та повертати квитки, задаючи кількість та категорію. Кожен користувач має деяку суму на рахунку. Підчас повернення квитка повертається не вся вартість повністю, а тільки деякий відсоток. Адміністратор може вносити інформацію про нові концерти, редагувати кількість доступних квитків.

1.3.2.28 Завдання:

а) У файлі міститься перелік слів, відділених одне від одного комами. Створити максимально можливий за площею прямокутник з даних слів таким чином, щоб прямокутник був сформований зі слів, розташованих за спіраллю таким чином, що остання літера попереднього слова та перша літера поточного слова співпадають. Слова можуть бути розташовані у довільному порядку. Наприклад:

a b c d
j k l e
i h g f

Слова abcd, def, fghi, ij, jkl.

В якості спрощення можна передбачити, що слова мають невелику довжину.

б) Перевірка знань на отримання водійського посвідчення. Кожен тест відповідає певній темі і складається з набору запитань. Має надаватися можливість додавання запитань до тесту, формування білету за темою. Для кожного запитання може визначатися кількість

варіантів відповідей і вірних варіантів, безпосередньо самі варіанти відповідей, а також вірні відповіді. Формування білету за темою відбувається шляхом випадкового вибору 20 запитань зі всіх доступних за темою.

1.3.2.29 Завдання:

а) У файлі містяться дані про товари на складах, що включають: назва складу, адреса, відстань до складу, вартість постачання, назва товару, вартість одиниці, кількість одиниць. Прочитати дані з файлу та сформувати з них зв'язний список. Реалізувати пошук за списком найдешевшого товару, заданого назвою.

б) Клієнт може мати декілька рахунків у банку. Менеджер може додавати, переглядати, редагувати (вносити кошти на рахунок) та вилучати рахунки. Клієнт може переглядати стан всіх своїх банківських рахунків (після авторизації), сплачувати за послуги, переглядати перелік операцій, що вже відбулись.

1.3.2.30 Завдання:

а) На шахівниці розташовані фігури, серед яких може бути кінь, пішак, тура. При чому може бути декілька відповідних фігур на дошці. Розташування фігур на дошці задано в файлі. Користувач визначає одну з клітинок. Визначити, яка з фігур і за яку найменшу кількість кроків зможе розташуватися в заданій клітинці.

б) Керування нотатками. Користувачі можуть додавати власні нотатки, редагувати існуючі або вилучати їх. Кожна нотатка характеризується назвою, текстом та категорією. Користувачі можуть переглядати тільки власні нотатки: всі загалом, за категоріями, виконувати пошук нотаток за словами з назви або тексту.

1.3.3. Виконати тестування розробленого програмного забезпечення.

1.3.4. Оформити звіт.

1.3.5. Відповісти на контрольні запитання.

1.4. Зміст звіту

1.4.1. Мета роботи.

1.4.2. Завдання до роботи.

1.4.3. Тексти розробленого програмного забезпечення.

1.4.4. Результати тестування: вхідні дані та результати роботи програми.

1.4.5. Перелік ризиків з описом та відповідних їм засобів управління.

1.4.6. Висновки, що відображають особисто отримані результати виконання роботи, їх критичний аналіз та порівняння парадигм програмування.

1.5. Контрольні запитання

1.5.1. Що таке системний аналіз?

1.5.2. Які принципи системного підходу?

1.5.3. Який інструментарій може використовуватись для розроблення програм мовою Python?

1.5.4. Які парадигми програмування підтримуються мовою Python?

1.5.5. Які принципи лежать в основі програмування мовою Python?

1.5.6. Для яких цілей може використовуватися мова Python?

1.5.7. Які основні типи даних мови програмування Python?

1.5.8. Чим відрізняється динамічна типізація від статичної?

1.5.9. Чим відрізняється імперативна парадигма програмування від декларативної?

1.5.10. Які типи даних у Python є незмінними та яким чином змінити їх значення?

1.5.11. Які типи даних у Python можуть змінюватись?

1.5.12. Яким чином виконати форматування рядка?

1.5.13. Що таке перелік та чим він відрізняється від інших типів даних?

1.5.14. Які дії можна виконувати над переліками?

1.5.15. Що таке множина та яким чином її визначити?

1.5.16. Що таке словник та чим він відрізняється від інших типів даних?

1.5.17. Які дії можна виконувати над словниками?

1.5.18. Що таке кортеж та у яких випадках його необхідно використовувати?

1.5.19. Яким чином визначається цикл з постумовою у мові Python?

1.5.20. Яким чином визначити у Python аналог оператора switch у C/C++?

1.5.21. Яким чином визначити рекурсію в Python?

1.5.22. Чим відрізняється синтаксис мови Python від C/C++?

1.5.23. Яким чином виконується робота з файлами в Python?

1.5.24. Які існують області видимості в Python?

1.5.25. Яким чином згенерувати перелік або словник?

1.5.26. Яким чином виконується обробка виключень?

1.5.27. Яким чином застосовується системний підхід під час створення коду?

2. ЛАБОРАТОРНА РОБОТА № 2 ВИКОРИСТАННЯ ФУНКЦІОНАЛЬНОГО ПРОГРАМУВАННЯ ДЛЯ ПОБУДОВИ ПРОГРАМНОЇ СИСТЕМИ

2.1. Мета роботи

2.1.1 Ознайомитись з основними особливостями функціональної парадигми програмування та особливостями її реалізації в мові програмування Python.

2.1.2 Навчитися розробляти програми мовою програмування Python на основі використання парадигми функціонального програмування.

2.2. Короткі теоретичні відомості

Функціональне програмування – це стиль написання програм через створення набору функцій.

Функціональне програмування є однією з парадигм, які підтримує мова програмування Python як мультипарадигмальна мова програмування.

Поняття функції використовується як в функціональному програмуванні, так і за імперативної парадигми.

Розглянемо приклад функції, написаної не у функціональному стилі.

Python

```
val = 0
def incfun1():
    global val
    val += 1
```

Аналогічна функція, створена в функціональному стилі має наступний вигляд:

Python

```
incfun2 = lambda a : a+1
```

До функцій, які забезпечують реалізацію базових принципів функціонального програмування в мові Python, належать:

– map;
– reduce;
– filter.

2.3. Завдання на лабораторну роботу

2.3.1. Ознайомитись з теоретичними відомостями, необхідними для виконання роботи.

2.3.2. Розробити програмне забезпечення у відповідності з індивідуальним завданням, узгодженим з викладачем. Для розроблення програмного забезпечення використовувати тільки парадигму функціонального програмування.

Для **розв'язання завдання а)** використовувати тільки виклики функцій, визначення анонімних функцій на основі викликів інших функцій, рекурсію (тобто заборонено використовувати цикли, умовні оператори, оператори присвоювання, оператори контролю окрім return). Для **розв'язання завдання б)** окрім того обов'язково використовувати спеціальні функції, що реалізують можливості функціонального програмування (map, reduce тощо), віддавши їм перевагу над всіма іншими способами реалізації.

Після цього розв'язати завдання а) другим способом, застосовуючи парадигму імперативного програмування.

2.3.2.1 Завдання для варіанта № 1.

а) З заданої послідовності чисел сформувати нову, в якій на кожній позиції розташувати середньогеометричне значення всіх елементів початкової послідовності окрім елемента, розташованого на відповідній позиції.

б) У текстовому файлі містяться пари слів або словосполучень, кожна з яких складається зі слова (словосполучення) українською мовою, тире та відповідного слова (словосполучення) англійською

мовою. Сформувати словник з даного набору слів. Виконати переклад тексту, заданого користувачем, з української мови на англійську. Для спрощення знаки пунктуації не враховувати. Враховувати, що в словнику зустрічаються як слова, так і словосполучення, а за наявності такого словосполучення у тексті віддавати йому перевагу.

2.3.2.2 Завдання для варіанта № 2.

а) Модифікувати заданий перелік чисел, вилучивши з нього всі від'ємні числа, а всі додатні розташувати у зворотному порядку.

б) Інформація про студентів міститься в файлі у наступному вигляді: прізвище, ім'я, по батькові, група, спеціальність, дисципліна, викладач, дата іспиту або заліку, кількість балів. Сформувати набір відповідних структур даних з файлу і вивести перелік студентів, які мають більше двох заборгованостей, у порядку зменшення їх кількості заборгованостей. У випадку рівної кількості порівнювати за середнім балом, отриманим студентом за даними заборгованостями у відомостях. А у випадку рівності – за номером групи.

2.3.2.3 Завдання для варіанта № 3.

а) Виділити з заданого повідомлення найдовшу послідовність, що складається з літер, які входять в заданий рядок.

б) Для заданого переліку чисел виконати нормування елементів.

2.3.3.4 Завдання для варіанта № 4.

а) Написати програму шифрування та дешифрування текстового повідомлення. Шифрування виконати наступним чином: розташувати спочатку всі символи, розташовані на парних позиціях, а потім у зворотному порядку всі символи, розташовані на непарних позиціях.

б) Сформувати з заданого переліку новий перелік таким чином, що на кожній його позиції знаходиться мінімальне значення серед всіх елементів початкового переліку, починаючи з першого елементу і закінчуючи поточною позицією.

2.3.3.5 Завдання для варіанта № 5.

а) Промоделювати деяку гру. Гра відбувається на прямокутному полі. Гравець на початку розташовується у правому верхньому куті. Гравець вміє переміщуватися тільки вліво та вниз. Кожна клітинка поля має деяку кількість очок. Дані задані у вигляді файлу. Визначити спосіб переміщення з початкового положення гравця на вказану користувачем клітинку таким чином, щоб він

призвів до отримання найбільшої кількості очок (розраховується як сума очок, зібраних протягом всього шляху на кожній клітинці).

б) Задано словник, у якому ключем є слово, а значенням – тлумачення слова. Сформувати два переліки, перший з яких складається зі слів словника, у тлумачення яких входить тлумачне слово, а другий – зі слів, у тлумаченні яких тлумачне слово не міститься. Кожен з переліків відсортувати так, щоб слова розташовувались у відповідності з алфавітним порядком відповідних їм тлумачень.

2.3.3.6 Завдання для варіанта № 6.

а) Виконати шифрування та дешифрування деякого повідомлення. Шифрування виконувати за допомогою ключа (*A*, *B*), сформованого з двох переліків *A* і *B*. Перелік *B* задає символи, за допомогою яких шифруються відповідні символи з переліку *A*.

б) У заданій послідовності цілих чисел визначити кількість від'ємних і додатних чисел. Сформувати нову послідовність з даної, підносячи елементи початкової у ступінь, що дорівнює відношенню кількості додатних чисел до кількості від'ємних, округленому до цілого

2.3.3.7 Завдання для варіанта № 7.

а) Виділити з заданого тексту найдовшу послідовність символів, що розташовані поспіль в алфавітному порядку.

б) У Appstore формується перелік користувачів, що оформили попереднє замовлення на купівлю iPhone 7. Перелік включає номер замовлення, прізвище, ім'я, AppleID, номер кредитної картки, дату замовлення. Даний перелік представлено у вигляді файлу. Вилучити всі повторні замовлення користувачів з файлу. Існує також файл з попередніми замовленнями iPhone6. Визначити, хто з користувачів представлений у обох замовленнях. Сформувати третій файл, у який вивести дані про таких користувачів.

2.3.3.8 Завдання для варіанта № 8.

а) Сформувати з двох заданих переліків чисел однакової довжини, відсортованих за зростанням, третій. Елементи сформованого переліку повинні бути також відсортовані за зростанням (без окремого сортування об'єднаного переліку).

б) З даного текстового файлу визначити слово, яке має найбільшу кількість голосних звуків.

2.3.3.9 Завдання для варіанта № 9.

а) У двомірному масиві, що містить дійсні числа, знайти суму елементів, що розташовані нижче головної та побічної діагоналей масиву.

б) Користувач визначає текст. Визначити набір літер (вважати, що всі слова в тексті належать до однієї мови, але різний заданий текст може належати до різних мов), які зустрічаються найменшу кількість разів серед кожного п’ятого слова тексту. Обчислити кількість таких літер.

2.3.3.10 Завдання для варіанта № 10.

а) Змінити порядок символів у тексті, розташувавши їх у порядку, заданому переліком, який містить числа, що відповідають новим позиціям символів у тексті. При цьому враховувати, що перелік може мати довжину, меншу за текст: у такому випадку розділяти текст на порції довжиною рівною довжині переліку і у кожній порції порядок змінювати відповідно до переліку.

б) Кожен з трьох переліків складається зі слів. Окрім того задано слово-правило. Потрібно перевірити, яка кількість слів з кожного переліку складається тільки з літер слова-правила і більше ніяких інших. Визначити перелік, який містить найбільше таких слів.

2.3.3.11 Завдання для варіанта № 11.

а) У текстовому повідомленні виділити слова, що входять до його складу, та визначити їх кількість, не використовуючи додаткові спеціалізовані функції для виділення слів. Враховувати, що слова відділяються одне від одного не тільки пробілами, але й будь-якими знаками пунктуації.

б) Перетворити заданий набір чисел на новий таким чином, щоб на кожній позиції розташувати середньоарифметичне значення всіх елементів початкового набору окрім елемента, розташованого на відповідній позиції.

2.3.3.12 Завдання для варіанта № 12.

а) Визначити кількість однакових за значенням елементів заданих двох матриць, що розташовані на їх побічних діагоналях.

б) У файлі містяться дроби, записані у форматі «чисельник/знаменник». Користувач з клавіатури задає число *N*. Знайти у даному файлі всі правильні нескоротні дроби, знаменники

яких не більше *N*. Визначити всі можливі правильні нескоротні дроби, знаменники яких менше *N* і які не зустрічаються в даному файлі.

2.3.3.13 Завдання для варіанта № 13.

а) Визначити кількість входжень кожного символу в заданий текст і вивести їх (символи і кількість входжень) на екран, відсортувавши в порядку зменшення кількості входжень.

б) Для заданої матриці та номера її рядка визначити кількість рядків матриці, які мають більше одного спільного елемента з рядком матриці під заданим номером.

2.3.3.14 Завдання для варіанта № 14.

а) Задано прямокутне поле, розмір якого задається користувачем. Робот розташовується у лівому верхньому куті. Робот вміє переміщатися тільки вправо та вниз. Визначити кількість способів, якими робот може переміститися в точку з заданими координатами. Вивести на екран кількість способів та шлях для кожного способу (наприклад, одиницями позначити рух робота, а всі інші клітинки – нулями).

б) Задано два масиви чисел *A* і *B*. Обчислити суму попарних множень елементів масивів, попередньо підносячи елементи масиву *A* у другий ступінь, зсуваючи перед цим елементи таким чином, щоб не використовувати в обчисленнях від'ємні значення.

2.3.3.15 Завдання для варіанта № 15.

а) Визначити в заданому користувачем тексті найдовшу послідовність символів, які не повторюються між собою.

б) Змінити набір чисел, вилучивши з нього всі додатні числа, а всі інші розташувати у зворотному порядку, початково перетворивши кожне значення елемента *x* наступним чином:

$$\sin^3(x+2).$$

2.3.2.16 Завдання для варіанта № 16.

а) Сформувати з заданого переліку $A=\{a_i\}, i=1...n$ новий перелік $B=\{b_i\}, i=1...n$ у відповідності з виразом:

$$b_i = \max_i^n a_i .$$

б) З даного текстового файлу визначити слово або слова, що мають найменшу кількість літер з заданої користувачем послідовності.

2.3.2.17 Завдання для варіанта № 17.

а) Користувачі формують слова з літер заданого слова TW, приймаючи участь у грі. Задано перелік слів, який починається зі слова TW, після чого слідують пари слів від першого та другого гравця <TW, w1, w2, w1, w2, …>. Перевірити та вивести на екран всі слова, які складені вірно з літер TW. Визначити, хто з гравців переміг.

б) Для кожного символу задано символ, за допомогою якого відбувається його шифрування. Реалізувати шифрування та дешифрування тексту, що зберігається в файлі. Якщо для якогось символу не задано спосіб шифрування, залишити його без змін.

2.3.2.18 Завдання для варіанта № 18.

а) У заданій послідовності чисел підрахувати кількість пар (послідовно розташованих чисел), представлених додатними значеннями, та кількість пар, значення елементів яких мають протилежний знак. Вивести повідомлення про те, кількість яких пар переважає.

б) Користувач визначає два рядки. Сформувати з них третій рядок, в якому розташувати послідовно літери першого та другого рядка на відповідних позиціях. Тобто спочатку перша літера першого рядка, потім – другого, далі – друга першого, друга другого тощо. При цьому попередньо рядки обробити та зсунути літери, звільнившись в першому рядку від голосних, а в другому – від приголосних.

2.3.2.19 Завдання для варіанта № 19.

а) Вивести всі коректні комбінації пар круглих дужок, які можна сформувати з *n* дужок, що закриваються і відкриваються. Наприклад, коректна комбінація – (()()), некоректна – ((()))(. Кількість дужок задається користувачем.

б) Визначити кількість слів у тексті, що зберігається в файлі, та довжину найкоротшого слова. Слова відділяються одне від одного не тільки пробілами, але й будь-якими знаками пунктуації.

2.3.2.20 Завдання для варіанта № 20.

а) У заданому двомірному масиві знайти найбільше значення локального максимуму. Локальним максимумом є елемент, значення якого більше значення всіх його сусідів.

б) Два переліки містяться в окремих файлах. Кожен з переліків складається з рядків. Вилучити з кожного переліку повторні входження кожного значення. Визначити елементи, які входять в обидва переліки, вивести їх на екран та обчислити їх кількість.

2.3.2.21 Завдання для варіанта № 21.

а) Елементи двомірного масиву є дійсними додатними числами. Розмірність масиву та значення його елементів задано користувачем. Обчислити суми елементів головної та побічної діагоналей масиву. Визначити, чи є отримані суми взаємно простими, якщо їх значення округлити до цілого.

б) У заданому текстовому файлі, що містить текст різними мовами, виділити кожне друге слово та знайти в них літеру, що найчастіше зустрічається (сумарно у всіх таких словах).

2.3.2.22 Завдання для варіанта № 22.

а) Виділити з заданого переліку чисел послідовність найдовшої довжини, що складається з елементів, розташованих поспіль, сума яких не перевищує задане значення.

б) Тлумачний словник визначає для кожного слова його тлумачення. Вивести на екран переліки, перший з яких складається зі слів словника, у тлумачення яких входить тлумачне слово, а другий – зі слів, у тлумаченні яких тлумачне слово не міститься. Кожен з переліків відсортувати так, щоб слова розташовувались у відповідності з порядком відповідних їм тлумачень, повністю протилежним алфавітному.

2.3.2.23 Завдання для варіанта № 23.

а) Вилучити з заданої послідовності чисел всі послідовності елементів, що складаються з більше ніж двох розташованих поспіль і одночасно розташованих за збільшенням значень елементів.

б) Користувач визначає пошуковий запит. Необхідно виконати пошук, використовуючи файл, у якому міститься перелік файлів, кожний з яких містить деякий текст. Оцінити відношення кількості входжень кожного слова із запиту в кожний файл до кількості слів у даному файлі. Оцінити відношення кількості входжень пар слів (слова розташовані поспіль) з пошукового запиту в кожний файл до кількості таких пар у файлі і т.д. Обчислити добуток даних оцінок та визначити на основі даного показника текст, який найбільше відповідає запиту.

2.3.2.24 Завдання для варіанта № 24.

а) Для заданого переліку чисел виконати нормування елементів.

б) З визначеного користувачем тексту, яке містить слова різними мовами (українською, російською, англійською), визначити найдовше слово англійською мовою. При чому деякі слова можуть містити літери з різних абеток. При визначенні враховувати тільки ті, що містять літери однієї абетки.

2.3.2.25 Завдання для варіанта № 25.

а) У заданому переліку чисел, рухаючись по ньому з початку до середини, замінити всі елементи, розташовані на непарних позиціях, на елементи, розташовані на відповідних позиціях з кінця переліку, а всі елементи, розташовані на парних позиціях, – на середнє арифметичне між кожним таким елементом та елементом, розташованим на відповідній позиції з кінця переліку.

б) Визначити кількість входжень кожного символу в заданий текст і вивести їх (символи і кількість входжень) на екран, відсортувавши в порядку зменшення кількості входжень. Враховувати, що в тексті можуть міститися не тільки літери, а літери можуть належати до різних алфавітів.

2.3.2.26 Завдання для варіанта № 26.

а) У заданій послідовності цілих чисел визначити кількість парних і непарних чисел. Сформувати нову послідовність з даної, підносячи елементи початкової у ступінь, що дорівнює відношенню кількості парних чисел до кількості непарних, округленому до цілого.

б) Сформувати з заданого переліку новий перелік таким чином, що на кожній його позиції знаходиться середнє значення серед всіх елементів початкового переліку, починаючи з наступної позиції за поточною і закінчуючи останнім елементом.

2.3.2.27 Завдання для варіанта № 27.

а) Сформувати одномірний масив з заданого двомірного масиву, що містить цілі числа, наступним чином: кожен елемент одномірного масиву має дорівнювати сумі парних додатних елементів відповідного стовпця двомірного масиву.

б) Користувач визначає два рядки. Сформувати з них третій рядок, в якому розташувати послідовно літери першого та другого рядка на відповідних позиціях. Тобто спочатку перша літера першого рядка, потім – другого, далі – друга першого, друга другого тощо. При

цьому попередньо рядки обробити та зсунути літери, звільнившись в першому рядку від голосних, а в другому – від приголосних.

2.3.2.28 Завдання для варіанта № 28.

а) Користувачем задано довільну послідовність *A*. Перевірити, чи вірно, що дану послідовність *A* можна сформувати з деякої послідовності *B*, повторюючи її деяку кількість разів. Послідовність *B* визначається з послідовності *A* відкиданням деякої кількості її елементів, що йдуть поспіль, з кінця. Вивести на екран відповідне повідомлення.

б) Визначити кількість слів у тексті, що зберігається в файлі, та довжину найдовшого слова. Слова відділяються одне від одного не тільки пробілами, але й будь-якими знаками пунктуації.

2.3.2.29 Завдання для варіанта № 29.

а) Визначити у заданій послідовності чисел найдовшу підпослідовність, яка складається з розташованих поспіль чисел, що формують арифметичну прогресію.

б) З даного текстового файлу визначити слово, яке має найбільшу кількість приголосних звуків.

2.3.2.30 Завдання для варіанта № 30.

а) Для заданого переліку чисел перевірити, чи є він низхідним, якщо всі від'ємні значення піднести у квадрат. Якщо таке твердження вірне, то повернути суму чисел з переліку, а інакше – добуток.

б) Задано словник, у якому ключем є слово, а значенням – тлумачення слова. Сформувати два переліки, перший з яких складається зі слів словника, у тлумачення яких входить тлумачне слово, а другий – зі слів, у тлумаченні яких тлумачне слово не міститься. Кожен з переліків відсортувати так, щоб слова розташовувались у відповідності з алфавітним порядком відповідних їм тлумачень.

2.3.3. Проаналізувати ризики, які виникли в процесі реалізації програм на основі функціональної й імперативної парадигм і в загальному випадку в процесі мультипарадигмального програмування, та які можуть виникнути в процесі експлуатації відповідних реалізацій програмного забезпечення. Розглянути як можна більше проблем та відповідних ризиків.

2.3.4. Виконати тестування розробленого програмного забезпечення.

2.3.5. Оформити звіт.
2.3.6. Відповісти на контрольні запитання.

2.4. Зміст звіту

2.4.1. Мета роботи.
2.4.2. Завдання до роботи.
2.4.3. Тексти розробленого програмного забезпечення.
2.4.4. Результати тестування: вхідні дані та результати роботи програми.
2.4.5. Перелік ризиків з описом та відповідних їм засобів управління.
2.4.6. Висновки, що відображають особисто отримані результати виконання роботи, їх критичний аналіз та порівняння парадигм програмування.

2.5. Контрольні запитання

2.5.1. Який інструментарій може використовуватись для розроблення програм мовою Python?
2.5.2. Які парадигми програмування підтримуються мовою Python?
2.5.3. Які принципи лежать в основі програмування мовою Python?
2.5.4. Для яких цілей може використовуватися мова Python?
2.5.5. Які основні типи даних мови програмування Python?
2.5.6. Чим відрізняється динамічна типізація від статичної?
2.5.7. Чим відрізняється імперативна парадигма програмування від декларативної?
2.5.8. Які типи даних у Python є незмінними та яким чином змінити їх значення?
2.5.9. Які типи даних у Python можуть змінюватись?
2.5.10. Яким чином виконати форматування рядка?
2.5.11. Що таке перелік та чим він відрізняється від інших типів даних?
2.5.12. Які дії можна виконувати над переліками?
2.5.13. Що таке множина та яким чином її визначити?

2.5.14. Що таке словник та чим він відрізняється від інших типів даних?

2.5.15. Які дії можна виконувати над словниками?

2.5.16. Що таке кортеж та у яких випадках його необхідно використовувати?

2.5.17. Яким чином визначається цикл з постумовою у мові Python?

2.5.18. Яким чином визначити у Python аналог оператора switch у C/C++?

2.5.19. Яким чином визначити функцію у Python?

2.5.20. Чи може кількість параметрів у функції змінюватись?

2.5.21. Які особливості функцій у мові Python порівняно з мовою C++?

2.5.22. Яким чином визначити рекурсію в Python?

2.5.23. Чим відрізняється синтаксис мови Python від C/C++?

2.5.24. Яким чином виконується робота з файлами в Python?

2.5.25. Які існують області видимості в Python?

2.5.26. Яким чином згенерувати перелік або словник?

2.5.27. Що таке анонімна функція?

2.5.28. Чим відрізняється робота з функціями в мовах Python та C/C++?

2.5.29. Яким чином реалізується парадигма функціонального програмування в Python?

2.5.30. Яким чином виконується обробка виключень?

2.5.31. Яким чином визначити потрібну парадигму при створенні програмної системи?

2.5.32. В яких випадках має переваги функціональне програмування при створенні програмних систем?

3. ЛАБОРАТОРНА РОБОТА № 3 СИСТЕМНИЙ АНАЛІЗ ПІД ЧАС ПОВТОРНОГО ВИКОРИСТАННЯ КОДУ

3.1. Мета роботи

Навчитися застосовувати принципи системного аналізу під час повторного використання коду, навчитися розробляти ігрові додатки на основі використання існуючих рушіїв з врахуванням результатів системного аналізу проблеми.

3.2. Короткі теоретичні відомості

Повторне використання коду в процесі розробки програмного забезпечення пов’язано з ризиками. Ризики можуть виникати зокрема через те, що на вивчення коду може не відводитись достатньо часу, а також через те, що відсутнє розуміння області застосування коду. Для абсолютного виконання всіх поставлених завдань за повторного використання коду необхідно провести детальний аналіз коду, виділити ключові для повторного використання частини. Окрім того даний процес неможливий без врахування зв’язків всередині створюваної системи, а також між системою та рушієм. Відповідно виконання системного аналізу є обов’язковим для розроблення програмної системи на основі повторного використання коду.

Ігровий рушій – це центральний програмний компонент комп’ютерних ігрових додатків з графікою, що обробляється в реальному часі.

Рушій зазвичай має наступні підсистеми:

– графічна підсистема;

– підсистема вводу;

– звукова підсистема;

– системне ядро.

Існує багато ігрових рушіїв, які можуть бути застосовані в процесі розробки мобільних ігор або браузерних ігор. Однак вибір має бути достатньо обґрунтованим, з урахуванням всіх ризиків та виконаним після достатньо детального аналізу.

Pygame – це набір кросплатформних Python-модулів, спроєктованих для створення комп'ютерних ігор. В основі Pygame лежить крос-платформна мультимедійна бібліотека SDL.

Pygame передбачає попереднє встановлення перед тим, як його можна буде імпортувати в програмі мовою Python як звичайний пакет. Для цього найзручніше застосовувати систему керування пакунками pip, виклик якої можна виконати наступним чином:

cmd

```
python3 -m pip install pygame –user
```

Pygame надає наступні модулі:

– cdrom – керування пристроями ком пакт-дисків та відтворення звуку;

– cursors – завантаження зображень курсору;

– display – керування вікном або екраном;

– draw – малювання графічних примітивів на поверхні (клас Surface);

– event – керування подіями та чергою подій;

– font – створення та відображення шрифтів TrueType;

– image – збереження та завантаження зображень;

– joystick – керування джойстиками;

– key – керування клавіатурою;

– mouse – керування маніпулятором типу «миша»;

– movie – відтворення відео;

– sndarray – керування звуками за допомогою класу Numeric;

– surfarray – керування зображеннями за допомогою класу Numeric;

– time – керування таймерами;

– transform – зміна розмірів, обертання та зміна орієнтації зображення [12].

Розглянемо приклад для створення анімації м'яча за допомогою використання можливостей Pygame [12]:

Python

```
import sys, pygame
pygame.init()

size = width, height = 320, 240
speed = [2, 2]
black = 0, 0, 0

screen = pygame.display.set_mode(size)

ball = pygame.image.load("ball.bmp")
ballrect = ball.get_rect()

while 1:
    for event in pygame.event.get():
        if event.type == pygame.QUIT: sys.exit()

    ballrect = ballrect.move(speed)

    if ballrect.left < 0 or ballrect.right > width:
        speed[0] = -speed[0]
    if ballrect.top < 0 or ballrect.bottom > height:
        speed[1] = -speed[1]

    screen.fill(black)
    screen.blit(ball, ballrect)
    pygame.display.flip()
```

3.3. Завдання на лабораторну роботу

3.3.1. Ознайомитись з теоретичними відомостями, необхідними для виконання роботи.

3.3.2. Визначити функціональні вимоги до програмного забезпечення у відповідності з індивідуальним завданням, узгодженим з викладачем:

3.3.2.1 Завдання для варіанта № 1.

У грі в сніжки беруть участь два гравці, дії одного з яких симулюються програмно. Рухом та кидками другого гравця керує користувач. Потрапляння сніжки в гравця призводить до того, що він виключається на деякий час з дії. Гра відбувається в двох режимах: за заданий час або з заданою кількістю сніжок. У кожного гравця є свій рівень енергії. Енергія зменшується при ураженні сніжкою та збільшується при потраплянні сніжок у супротивника. Переможцем вважається відповідно гравець, який влучив у свого супротивника найбільше. Зниження поточного рівня енергії нижче мінімального рівня призводить до передчасного програшу.

3.3.2.2 Завдання для варіанта № 2.

Гра в більярд виконується за наступними правилами. Частина куль розташовується на столі випадковим чином. Інші кулі розташовуються у формі піраміди. Кожна куля має певний колір і відповідну кількість очок, що нараховується у випадку потрапляння її в лузу. Усі кулі в піраміді мають однаковий колір. Гравці наносять удари білою кулею по іншим кулям. Переможцем вважається гравець, який набрав найбільшу кількість очок. Дії другого гравця моделюються програмно. Гра відбувається в двох варіантах. У першому гравці виконують удари по черзі. У другому гравець виконує удари, поки жодна з куль не потрапить у лузу.

3.3.2.3 Завдання для варіанта № 3.

Автоперегони відбуваються на одній з обраних користувачем трас, які мають різну складність. Рух декількох автомобілів симулюється програмно. Мета гри полягає в тому, щоб приїхати першим за найкоротший відрізок часу. Усі результати гравця зберігаються. На трасах розташовані також перепони, які призводять до уповільнення автомобіля або його затримки на певний час у випадку потрапляння на них. Аналогічні дії відбуваються у випадку зіткнення з іншими автомобілями. Виліт з траси призводить до втрати часу. Передбачити також підрахунок кількості штрафних очок, які нараховуються у випадку настання перерахованих випадків.

3.3.2.4 Завдання для варіанта № 4.

Гравець, яким керує користувач, виконує удари по воротам, які захищаються воротарем. Воротар рухається програмно визначеним шляхом. Користувач має забити якомога більшу кількість м’ячів за

відведений проміжок часу. На більш складному рівні ворота захищає також стінка з гравців, розташована на деякий відстані, або певна кількість гравців, які рухаються на деякій відстані програмно визначеним шляхом.

3.3.2.5 Завдання для варіанта № 5.

Гра імітує дику природу, де одні звірі їдять інших, а людина полює на звірів. Людина може вбити звіра шляхом пострілу. Завдання користувача, представленого одним зі звірів, – вполювати іншого звіра та не потрапити під постріл людини. За впольовану здобич сили збільшуються. Швидкість руху залежить від сил. У випадку потрапляння пострілу – сили зменшуються в залежності від місця потрапляння (або одразу призводять до смерті звіра у випадку потрапляння в серце). Гра відбувається протягом відведеного періоду часу.

3.3.2.6 Завдання для варіанта № 6.

Користувач керує деяким предметом, який може стрибати над поверхнею. На поверхні розташовуються предмети, одні з яких завдають шкоди, знімаючи з гравця певну кількість очок і уповільнюючи його рух або призводячи до закінчення гри, а інші навпаки додають очки при потраплянні на них. Користувач проходить весь простір зліва направо. Мета полягає в тому, щоб в результаті набрати максимальну кількість очок. Результати гри зберігаються.

3.3.2.7 Завдання для варіанта № 7.

М’ячі падають згори екрана вниз, де і залишаються, розташовуючись на інших м’ячах або на нижній лінії екрана. М’ячі мають різні кольори з деякого набору. Вибір кольору відбувається випадковим чином. Якщо один на одному або на одній лінії розташовані м’ячі одного і того самого кольору, то вони зникають. Швидкість руху м’ячів з кожним разом збільшується. Завдання полягає в тому, щоб направляти кожний м’яч необхідним чином, уникаючи розташування м’ячів одного на одному таким чином, що одна з пірамід дорослe догори екрану. Результат підраховується за двома показниками: пройденим часом та кількістю спалених м’ячів. Користувач обирає перед початком гри, який з варіантів підрахунку результатів буде використовуватись.

3.3.2.8 Завдання для варіанта № 8.

Гра полягає в тому, щоб рухаючи кошик у руках гравця (нижче, вище підіймаючи руки, вправо, вниз), ловити предмети, що падають згори. Деякі з даних предметів збільшують кількість очок, а інші – зменшують. З часом швидкість руху даних предметів пришвидшується. Необхідно набрати максимальну кількість очок. Гра закінчується або коли кількість предметів, що впала згори, перевищила деякий поріг, або коли кількість предметів в кошику перевищала максимально допустиме значення.

3.3.2.9 Завдання для варіанта № 9.

Гірськолижник, яким керує користувач, бере участь у змаганнях зі слалому. Траса позначена жердинами (ворота), у простір між якими має потрапити гірськолижник. За влучне проходження воріт користувач отримує бали. Якщо користувач збиває або торкається жердин, то бали відраховуються. Гірськолижник може збільшувати швидкість або зменшувати. У випадку руху з великою швидкістю на повороті він може вилетіти з траси. Результат визначає час проходження траси та кількість балів. Якщо кількість балів нижче деякого рівня, то час не зараховується.

3.3.2.10 Завдання для варіанта № 10.

Користувач керує космічним кораблем, який подорожує космосом. На його шляху трапляються планети, на яких знаходяться істоти. Корабель може доставляти їх з однієї планети на іншу (кожного потрібно доставити на деякі конкретні планети), отримуючи за це очки (кількість залежить від істоти). Корабель вміщує обмежену кількість істот. На шляху корабля трапляються інші кораблі, які можуть стріляти в нього. Гравець повинен їх знищити. Кожен їх влучний постріл призводить до зниження кількості очок. Гра завершується, коли кількість очок дорівнює 0 (спочатку дорівнює деякому додатному числу) або коли завершився відведений час.

3.3.2.11 Завдання для варіанта № 11.

Тренування з бейсболу відбувається наступним чином: один гравець кидає м’яч, а другий – його відбиває. Задача другого гравця – відбити м’яч у певну зону. В залежності від результату відбиття нараховуються бали першому та другому гравцям. Гра триває певну кількість спроб, яку може визначати користувач на початку.

3.3.2.12 Завдання для варіанта № 12.

Лабіринт містить об’єкти різної форми, розташовані на шляху гравця. Одні об’єкти повинні бути повністю зібрані гравцем для завершення гри, а інші він може або пересувати вперед на вільні місця, або вони заважають його руху, повністю закриваючи прохід (він може їх обійти іншим шляхом). Лабіринт обов’язково містить глухі кути. Задача гравця – пройти лабіринт, зібравши всі необхідні об’єкти повністю за найменший час.

3.3.2.13 Завдання для варіанта № 13.

На власному автомобілі гравець долає деяку нерівну трасу (містить гори, впадини тощо) зліва направо. Протягом руху на шляху автомобіля зустрічаються монети, які він може збирати. При цьому монети можуть розташовуватися дещо вище поверхні. Автомобіль в залежності від швидкості (прискорення) може або їхати по поверхні, або підскакувати над нею (в результаті долання вершин). За надвеликого прискорення та схилів, автомобіль може вилетіти з траси. В результаті успішного проходження траси підраховується кількість набраних монет.

3.3.2.14 Завдання для варіанта № 14.

Сноубордист долає трасу, що складається з набору гірок (один її бік веде вгору, інший вниз, відповідно підіймаючись, швидкість зменшується, а спускаючись, збільшується). В залежності від заданої початкової швидкості він може або подолати гірку, перестрибнувши на наступну, або ні, лишившись в межах поточної. В різних місцях над гіркою розташовуються предмети, які приносять бали. Схопивши такий предмет, сноубордист зменшує свою швидкість, але зменшує час проходження траси. Завдання полягає в тому, щоб подолати трасу якнайшвидше (з врахуванням балів).

3.3.2.15 Завдання для варіанта № 15.

Гравець представляє собою людину, що перетинає локацію зліва направо і протягом руху зустрічає перешкоди, кожна з яких представляє собою вертикальну межу з отвором. Завдання – потрапити в даний отвір, не торкаючись його меж. Кількість невдалих спроб обмежена. Для того щоб потрапити у отвір, гравець має підстрибнути (задаючи прискорення, траєкторію руху, і таким чином впливаючи на результат). Необхідно пройти локацію якомога швидше. Під час руху на деяких позиціях (на землі, вище) розташовуються

монети. Якщо зібрати їх у певній кількості, то їх можна обміняти на додаткове життя (невдалу спробу).

3.3.2.16 Завдання для варіанта № 16.

Танк знаходиться в приміщенні, план якого нагадує лабіринт. Окрім нього в даному лабіринті розташовуються ще декілька інших танків, а також предметів. Танк може стріляти, при чому снаряди відбиваються від стінок лабіринту (при цьому при потраплянні у предмети, снаряди та предмети знищуються), продовжуючи свій рух далі (протягом обмеженої кількості відбиттів, наприклад, – 3). У танк може потрапити задана максимальна кількість снарядів, після чого він вибухає. Завдання – керуючи одним з танків, перемогти суперників і залишитись останнім танком у приміщенні.

3.3.2.17 Завдання для варіанта № 17.

Гравець кидає м’яч у баскетбольний кошик, що рухається певним чином у деякій частині простору, повторюючи набір рухів. Задача гравця в тому, щоб потрапити у кошик максимальну кількість разів та набрати максимальну кількість очок. З часом швидкість руху кошика змінюється. Від швидкості руху кошика та дистанції до кошика залежить кількість очок, яку заробляє гравець при потраплянні в кошик м’ячем. Гравець може кидати м’яч у кільце тільки з обмеженої ділянки.

3.3.2.18 Завдання для варіанта № 18.

Деякий предмет (наприклад, м’яч) падає згори екрана, при чому місце початку падіння (горизонтальна позиція) та прискорення задаються користувачем. Користувач може направляти його подальший рух: вправо, вгору, вниз. На нижній лінії екрану розташовані різні предмети, розташування деяких з них може завадити просуванню предмета, яким керує користувач, або навпаки допомогти (використовуються в ролі батута). На шляху руху розташовуються предмети другого виду, потрапляння предмету користувача у них не змінює напрям та швидкість руху, але призводить до збільшення очок гравця. Набрана під час руху кількість очок визначає результат гравця.

3.3.2.19 Завдання для варіанта № 19.

На стенді стоїть деяка кількість об’єктів. Деякі з них рухаються. Користувач кидає в них набір предметів. Кожен з них має свої фізичні характеристики (прискорення, точність потрапляння). Кожен предмет

можна кинути тільки один раз. Кожен об’єкт на стенді у разі потрапляння в нього приносить певну кількість очок. Падаючи, предмети можуть збивати інші предмети, приносячи додаткову кількість очок. Необхідно набрати максимальну кількість очок. Результати користувача зберігаються.

3.3.2.20 Завдання для варіанта № 20.

Гра полягає в тому, щоб за деякий час наздогнати порушника. Користувач виступає в ролі поліцейського. Порушник втікає від поліцейського. На шляху їх руху трапляються різні предмети. Одні можуть виступати в ролі нерухомих перешкод (їх потрібно перестрибнути, або вони зупинять рух), а інші – потенційно рухомих (їх можна зсунути, знизивши швидкість, або кинути вперед). Дії порушника моделюються програмно. Поліцейський має уникати перешкод та наздогнати порушника.

3.3.2.21 Завдання для варіанта № 21.

Кінні перегони відбуваються на іподромі, де встановлені перешкоди. Коні мають перестрибувати встановлені перешкоди. Якщо кінь перешкоду не перестрибує, то він зупиняється, втрачаючи певний рівень сил. За зменшення сил нижче деякого рівня кінь сходить з дистанції. Мета перегонів – прийти на фініш першим. У перегонах беруть участь сім коней. Коні можуть зіштовхуватися, втрачаючи швидкість та сили. Випадковим чином коні можуть перешкоджати руху інший коней, зіштовхуючись.

3.3.2.22 Завдання для варіанта № 22.

Гравці розташовані з двох сторін від натягнутої на деякій висоті сітки, що розділяє ділянку для гри навпіл. Гра відбувається шляхом перекидання м’яча руками. Кожен гравець повинен перекинути сітку та потрапити м’ячем у ділянку супротивника, а не повз неї. За такого випадку він отримує очки за успішну дію. Окрім того супротивник може відбити м’яч на ділянку супротивника (тоді відповідно очки нараховуються вже йому у випадку успіху).

3.3.2.23 Завдання для варіанта № 23.

У акваріум поміщено три види рибок, які розрізняються розміром та агресивністю. Перші їдять тільки рослин. Другі їдять перших рибок, а треті – відповідно других. Через деякий час кількість перших рибок збільшується з деяким коефіцієнтом. Рибки третього виду рухаються активніше. Рибкою другого виду керує користувач. За

кожну з'їдену рибку енергія рибки користувача збільшується на деяку кількість. З часом кількість енергії рибки зменшується. Користувач може програти, якщо енергія його рибки зменшиться нижче деякого мінімального рівня або якщо її з'їсть рибка третього виду. В залежності від складності гри кількість рибок третього виду та швидкість руху всіх рибок збільшується.

3.3.2.24 Завдання для варіанта № 24.

Два боксери знаходяться в ринзі. Керування одним з них відбувається програмно, а іншим керує користувач. При зближенні на деяку відстань можна наносити удари. Кожен боксер може захищатися від удару, наносити удар або знаходитися в стандартній позиції. Знаходячись у стандартній позиції, боксер може нанести швидше удар ніж з позиції, в якій він захищається. Кожний точний удар зменшує енергію іншого боксера (в залежності від місця потрапляння на різну кількість). Від кількості енергії залежить швидкість руху. Якщо обидва боксери наносять удари, то точним вважається той удар, який досягає цілі першим, а другий удар не досягає цілі. Якщо енергія досягає нульового рівня, то бій завершується. Переможцем вважається той боксер, який після бою має більший рівень енергії.

3.3.2.25 Завдання для варіанта № 25.

На задньому плані розташована деяка піраміда, сформована з окремих предметів, що розташовуються один на одному. Користувач виконує постріл по даній піраміді. Місце, куди потрапляє куля, визначає кількість предметів, які впадуть з піраміди. Падіння кожного з предметів дає певну кількість балів гравцю (різну, наприклад, в залежності від рівня, на якому розташовується предмет у піраміді). Обрахувати кількість балів, отриману після пострілу. Надається деяка кількість пострілів. Кожний з наступних пострілів зменшує коефіцієнт, який використовується при обрахунку балів. Набрана кількість балів визначає результат.

3.3.2.26 Завдання для варіанта № 26.

Космічний корабель, яким керує гравець, мандрує космосом, знаходячи на своєму шляху різні об'єкти (предмети, літаючі апарати тощо). Корабель може або збивати такі об'єкти, або підбирати на борт, або уникати їх, при чому деякі об'єкти приносять очки за відповідну дію (наприклад, влучний постріл), а за іншої (наприклад, підбирання)

знімають очки. Окрім того деякі літальні апарати вміють виконувати постріли, яких має уникати гравець.

3.3.2.27 Завдання для варіанта № 27.

Два гравці грають у великий теніс. Спочатку один з гравців виконує подачу на протилежну частину, а далі кожен гравець наносить удари по м'ячу, перекидаючи його на протилежну частину. М'яч може потрапити в свою частину, повз неї, в протилежну частину, повз неї або в сітку. За кожний вдалий удар гравець отримує бали за тенісною системою і за нею ж фіксується переможець.

3.3.2.28 Завдання для варіанта № 28.

Внизу екрану розташовується рухома платформа, яка дозволяє відбивати вгору кулі, що по черзі рухаються згори вниз. Кулі мають різні кольори, що з часом повторюються. Відскік кулі залежить від прискорення наданого користувачем платформі. Якщо платформа не торкається кулі до її переміщення вниз, то куля зникає. При цьому відбита куля може бути направлена до куль, що залишились на екрані від минулих відбиттів. Якщо кулі однакового кольору, то торкання таких куль приносить бали. Бали тим вищі, чим більш схожі кольори.

3.3.2.29 Завдання для варіанта № 29.

Звірі рухаються в природному середовищі. Завдання гравця – вполювати звіра пострілом в деяку частину. Тільки у випадку вдалого пострілу йому нараховуються бали. У випадку невдалого – звір змінює свою швидкість, втікаючи, як і інші звірі, розташовані поруч. Гра відбувається протягом деякого заданого часу, за який необхідно набрати якомога більше балів.

3.3.2.30 Завдання для варіанта № 30.

На кожному кроці гри з'являється кулька певного кольору, яку гравець має направити в область, у якій розташовані інші, раніше направлені кульки. Користувач обирає напрям руху кульки, в результаті чого кулька розташовується в певній позиції, рухаючись за траєкторією, наштовхуючись на інші кульки. Після того, як кулька розташувалась на деякій позиції, користувач може або активувати режим згорання кульок, або нічого не робити. Після цього з'являється наступна нова кулька. У режимі згорання кульок усі кульки однакового кольору з останньою запущеною, які при цьому торкаються неї та одна одної (зліва, справа, згори, знизу, за діагоналлю), зникають з екрану, а інші, якщо розташовувались над

ними, займають їх позиції. Необхідно протриматися в грі якомога довше. Кульки поступово займають простір вікна.

3.3.3. Визначити систему об’єктів, які можуть використовуватися в заданій предметній області для реалізації індивідуального завдання. У процесі проєктування системи дотримуватися виконання вимоги її оптимальності для подальшої її реалізації за допомогою мільтипарадигмального підходу.

3.3.4. Визначити, яким чином застосувати дану систему об’єктів і реалізувати програмне забезпечення на основі повторного використання програмного коду, оцінивши при цьому різні можливі варіанти.

3.3.5. Розробити комп’ютерну гру за допомогою мови програмування Python і рушія Pygame у відповідності з індивідуальним завданням та спроєктованою системою об’єктів. Програмне забезпечення обов’язково має відповідати вимогам безпеки даних та програмного коду.

3.3.6. Визначити проблеми, які було ідентифіковано в процесі розроблення програмного забезпечення і такі, що можуть реалізуватись під час подальших етапів життєвого циклу програмного забезпечення, та відповідні їм засоби управління.

3.3.7. Оформити звіт.

3.3.8. Відповісти на контрольні запитання.

3.4. Зміст звіту

3.4.1. Мета роботи.

3.4.2. Завдання на роботу.

3.4.3. Концепція комп’ютерної гри.

3.4.4. Функціональні вимоги до програмного забезпечення.

3.4.5. Система об’єктів.

3.4.6. Текст програми.

3.4.7. Інтерфейс роботи з програмою в декількох режимах.

3.4.8. Результати роботи програми.

3.4.9. Таблиця, що містить назву проблеми, короткий опис, ситуація, в якій його може бути реалізовано, та перелік рішень, які можуть бути прийняті.

3.4.10. Висновки, що містять відповіді на контрольні запитання, а також відображають результати виконання роботи та їх критичний аналіз.

3.5. Контрольні запитання

3.5.1. Що таке ігровий рушій?

3.5.2. Для чого використовуються ігрові рушії?

3.5.3. Які частини комп'ютерної гри зазвичай дозволяє контролювати ігровий рушій?

3.5.4. Які ризики характерні для повторного використання коду?

3.5.5. Які ігрові рушії ви знаєте і які особливості вони мають?

3.5.6. Яка архітектура ігрового рушія Pygame?

3.5.7. Які бібліотеки має рушій Pygame?

3.5.8. Які функції надає рушій Pygame?

3.5.9. Чи варто використовувати код повторно?

3.5.10. Які найголовніші фактори успішного повторного використання програмного коду?

3.5.11. Які фактори необхідно враховувати при визначенні доцільності повторного використання коду для створення програмної системи?

3.5.12. Які засоби проєктування можна застосовувати для створення програмної системи на основі повторного використання коду?

3.5.13. Які елементи можуть розташовуватися на діаграмах класів?

4. ЛАБОРАТОРНА РОБОТА № 4
СТВОРЕННЯ ПРОГРАМНИХ СИСТЕМ

4.1. Мета роботи

4.1.1 Ознайомитись з принципами компонентно-орієнтованого програмування.

4.1.2 Навчитися застосовувати системний підхід для побудови програмного забезпечення.

4.2. Короткі теоретичні відомості

Розроблення та використання компонентів є основою компонентно-орієнтованого програмування. Дана парадигма програмування реалізується в багатьох технологіях, зокрема в мові програмування Java за допомогою технології JavaBeans, відомі також розповсюджені рішення CORBA, COM, SOAP. Платформа .NET реалізує компонентно-орієнтований підхід, забезпечуючи його використання для підтримуваних мов програмування.

В основі створення компонента .NET лежить клас .NET, наприклад:

C#

```
public class MyClass : IClassInterface
{
   public string GetMessage()
   {
      return "Just an example for a component";
   }
}
```

Цей клас створюється на основі організації відповідного інтерфейсу:

C#

```
public interface IClassInterface
{
  string GetMessage();
}
```

Компоненти .NET можуть належати збиранням на основі EXE (збирання застосувань) або DLL (бібліотечні збирання).

Для того щоб створити нове бібліотечне збирання C# у Visual Studio, необхідно скористатися меню File → New → Project... Після того як на екрані з'явиться діалогове вікно New Project, необхідно обрати Visual C# в якості типу проєкту, після чого обрати Windows, а у шаблонах – Class Library. Необхідно також задати розташування проєкту та ввести ім'я (наприклад, MyClassLib).

Після створення проєкту за допомогою меню можна додати новий клас, інтерфейс, компонентний клас тощо. Для цього призначений пункт меню Project → Add Class, який виводить на екран діалогове вікно Add New Item.

Для того щоб використати розроблений компонент у клієнтському застосуванні, необхідно виконати наступні дії:

а) створити новий проєкт Windows Application Form;

б) додати посилання на створене раніше збирання, обравши в меню Project → Add Reference…, після чого на екрані з'явиться діалогове вікно Reference Manager, у якому необхідно обрати створене збирання (обрати варіант Browse та вказати шлях до збирання);

в) додати директиву using для простору імен AssemblyDemo namespace;

г) використати в тексті програми розроблений компонент як звичайний клас, наприклад:

C#

```
using System;
using System.Windows.Forms;
using AssemblyDemo;
```

C#

```
partial class ClientForm : Form
{
  void OnClicked(object sender,EventArgs e)
  {
    IClassInterface obj = new MyClass( );
    string message = obj.GetMessage( );
    MessageBox.Show(message);
  }
}
```

4.3. Завдання на лабораторну роботу

4.3.1. Ознайомитись з теоретичними відомостями, необхідними для виконання роботи.

4.3.2. Розробити програмне забезпечення у відповідності з індивідуальним завданням, узгодженим з викладачем. Усі функції системи повинні бути реалізовані у вигляді необхідної кількості програмних компонентів .NET мовою програмування C#, для чого потрібно ще до реалізації забезпечити відповідний процес проєктування з обґрунтуванням отриманих результатів. Клієнтське застосування повинно використовувати функції з компонентів через програмні інтерфейси. Програмне забезпечення обов'язково має відповідати вимогам безпеки даних та програмного коду. Необхідні програмні компоненти спроєктувати на основі застосування системного підходу.

4.3.2.1 Завдання для варіанта № 1.

Розробити програмне забезпечення підтримки роботи служби таксі. Клієнти можуть замовляти таксі, обираючи автомобіль, маршрут, необхідний час прибуття. Визначається прогнозована вартість поїздки. У кінці поїздки таксист визначає фактичну тривалість поїздки, на основі якої визначається остаточна вартість послуги. Для кожного клієнта визначаються сумарні витрати на всі поїздки, від чого залежить дисконтна знижка. Менеджери визначають, за яким автомобілем закріплений який водій та чи доступний

відповідний автомобіль, чи ні (на деякий інтервал часу, наприклад, знаходиться в ремонті).

4.3.2.2 Завдання для варіанта № 2.

Розробити програмне забезпечення пошуку квитків на літак. Адміністратор визначає літаки, напрями їх слідування, кількість місць у літаку кожного типу, а також доступні місця на кожний конкретний літак. Менеджер може вносити дані про зайняті (викуплені) місця, а також вартість наявних місць. Система повинна виконувати пошук можливих варіантів перельотів, у тому числі з пересадкою, враховуючи обмеження за вартістю. Адміністратор може отримати інформацію про завантаженість літаків за напрямами та окремо літаків кожного типу за історією польотів.

4.3.2.3 Завдання для варіанта № 3.

Розробити програмне забезпечення ведення статистики футбольних змагань. Модератор вносить та змінює дані про команди, результати їх матчів, гравців, результати їх участі у матчах (кількість забитих голів, проведені на полі хвилини, червоні та жовті картки). Користувачі можуть переглядати статистику або обирати потрібну інформацію за визначеними ними гравцями, командами, переглядати найкращих в різних категоріях, де учасники відсортовані за параметрами (найкращі за позицією на полі, за результативністю, за кількістю проведених хвилин тощо).

4.3.2.4 Завдання для варіанта № 4.

Розробити програмне забезпечення проведення торгів. Користувачі протягом кожного дня формують заявки на продаж деякого товару у певній кількості за певною ціною. Інші користувачі формують власні заявки на купівлю даного товару, визначаючи кількість, вартість. У відведений адміністратором час протягом кожного дня (можливо декілька разів) система визначає переможців торгів та надсилає їм відповідні повідомлення. При цьому система визначає переможців за запропонованою найвищою ціною, але не менше визначеної продавцем, враховуючи, що переможців може бути декілька у відповідності з кількістю товару, що продається. Якщо заявку протягом поточного дня не задоволено, то вона переноситься на наступний день, якщо користувач її не скасував. Система повинна надавати можливість реєстрації та авторизації.

4.3.2.5 Завдання для варіанта № 5.

Розробити програмне забезпечення проведення вступної кампанії в університеті. Інформація про абітурієнтів вноситься в систему ними ж і включає прізвище, ім'я, по батькові, дату народження, середній бал документа про освіту, бали сертифікатів зовнішнього незалежного оцінювання (за трьома дисциплінами), додаткові бали та бажані спеціальності. Адміністратор вносить в систему інформацію про кількість місць на кожну спеціальність, відповідні сертифікати зовнішнього незалежного оцінювання і безпосередньо перелік спеціальностей. Система визначає абітурієнтів, які потрапили в прохідну частину переліку за кожною спеціальністю та надсилає їм повідомлення.

4.3.2.6 Завдання для варіанта № 6.

Розробити програмне забезпечення проведення тестування. Система повинна надавати можливість адміністратору створювати нове тестування, додавати запитання до визначеного тестування. Для запитання адміністратор визначає кількість варіантів відповідей і вірних варіантів, безпосередньо самі варіанти відповідей, а також вірні відповіді. Окрім того для кожного запитання користувач може визначити групу. Для кожного тестування адміністратор визначає кількість запитань, а також запитань кожної групи. Користувач може проходити тестування за вибраним тестом. Тест формується зі всієї кількості доступних запитань (за даним тестом), при чому необхідна кількість запитань кожної групи обирається випадковим чином. Система повинна надавати можливість реєстрації та авторизації.

4.3.2.7 Завдання для варіанта № 7.

Розробити програмне забезпечення проведення тендерів. Адміністратор визначає час проведення тендеру та обсяг робіт, які мають бути виконані, (або товарів, які мають бути закуплені). Користувачі пропонують власні заявки, зазначаючи обсяг та вартість. Перемагають заявки, у яких запропоновано найменшу вартість. При цьому обсяг замовлення може бути розподілений між декількома заявками. Експерти виставляють оцінку за кожною заявкою. Заявки, що набрали середню суму балів не менше заданої, можуть брати участь у тендері. Результати проведення тендеру надсилаються користувачам.

4.3.2.8 Завдання для варіанта № 8.

Розробити програмне забезпечення обліку роботи на підприємстві. Інформація про співробітників містить прізвище, ім’я і по батькові, посаду, тарифний розряд та заповнюється адміністратором. Адміністратор також визначає розмір погодинної оплати для кожного тарифного розряду, розмір податків і норму роботи на місяць. Секретар заповнює інформацію про облік роботи кожного співробітника за місяць, зазначаючи прізвище, ініціали, дату, час виходу на роботу, час завершення роботи. Понаднормовий робочий час сплачується подвійно. Визначити розмір заробітної платні кожного співробітника фірми з вирахуванням податків. Співробітники зареєстровані в системі та можуть отримати інформацію про нараховану заробітну платню.

4.3.2.9 Завдання для варіанта № 9.

Розробити систему, яка реалізує футбольний тоталізатор. Адміністратор визначає футбольні матчі, дати їх проведення та варіанти виграшів для можливих результатів (результат матчу, кількість забитих м’ячів). Користувачі роблять ставки, визначаючи результат та суму грошей. Система визначає результати матчів випадковим чином після настання відповідної дати. За отриманими результатами визначається виграш кожного учасника та надсилається йому відповідне повідомлення. Система повинна надавати можливість реєстрації та авторизації. Для кожного гравця накопичується певна сума за результатами його участі.

4.3.2.10 Завдання для варіанта № 10.

Розробити програмне забезпечення автоматизованого формування розкладу занять. Система повинна дозволяти користувачам формувати заявки, визначаючи номер групи, назву дисципліни, кількість занять на тиждень, викладачів і можливі аудиторії, та змінювати існуючі. Адміністратор визначає доступні аудиторії, університетські групи і викладачів. За результатами отриманих даних система повинна формувати розклад, розглядаючи при цьому декілька варіантів та обираючи з них той, якому відповідає менша кількість «вікон» у груп і викладачів (систему оцінювання визначити самостійно). Система повинна надавати можливість реєстрації та авторизації.

4.3.2.11 Завдання для варіанта № 11.

Розробити програмне забезпечення відправлення повідомлень. Кожен користувач може за її допомогою надсилати повідомлення, що включають e-mail одержувача, заголовок повідомлення, розмір повідомлення. E-mail відправника формується автоматично у відповідності з логіном користувача. Дата відправлення і час відправлення заповнюються системою також автоматично. Система повинна дозволяти користувачу переглядати власні повідомлення (надіслані та відправлені), сортуючи їх за часом відправлення та розподіляючи за даним показником на групи. Окрім того система повинна дозволяти користувачу виконувати пошук повідомлень та виводити повідомлення, відправлені в заданому часовому проміжку.

4.3.2.12 Завдання для варіанта № 12.

Розробити програмне забезпечення підтримки поціновувачів кінематографу. Адміністратори можуть додавати фільми та інформацію про них (режисер, актори, бюджет, дата випуску, жанр, синопсис тощо). Звичайні користувачі можуть переглядати інформацію про фільми, виконувати пошук фільмів, додавати власні відгуки про фільми та оцінювати їх. Для кожного фільму визначається середня оцінка за оцінками користувачів. Користувачі можуть також переглядати рейтинг найкращих фільмів за окремими жанрами або загалом.

4.3.2.13 Завдання для варіанта № 13.

Розробити програмне забезпечення замовлення комплексних обідів для працівників фірми. Кожен працівник може замовляти один із запропонованих обідів для доставляння на вказану дату, на кожен день або за розкладом (наприклад, декілька днів тижня перший варіант обіду, а всі інші – другий). Працівник може сплатити за обід одразу або обрати розрахунок у кінці місяця. Менеджери підрядника можуть визначати варіанти комплексних обідів, отримувати інформацію про всі замовлення, а також переглядати статистику про всі замовлення за обраний інтервал часу (місяць, рік).

4.3.2.14 Завдання для варіанта № 14.

Розробити програмне забезпечення проведення конкурсу. Користувачі подають заявки на конкурс. Адміністратор визначає оцінки, виставлені кожним експертом кожній заявці. Учасники, заявки яких отримали за результатами експертного оцінювання бали менше деякого заданого адміністратором рівня хоча б від двох експертів, до

конкурсу не допускаються. Система повинна розподілити суму фінансування між учасниками конкурсу в залежності від середнього балу експертного оцінювання. При цьому заявка не може отримати фінансування менше заданого мінімального рівня (всі такі заявки відхиляються від конкурсу).

4.3.2.15 Завдання для варіанта № 15.

Програмне забезпечення сплати за комунальні послуги. Адміністратори вносять інформацію в систему про тарифи на користування окремими послугами. Менеджери вносять інформацію про дані лічильників кожного користувача. Користувачі можуть переглядати власні рахунки, сплачувати їх, переглядати історію. Система інформує користувачів про заборгованість. За несплачені вчасно рахунки нараховується пеня. У випадку якщо заборгованість перевищує встановлений ліміт, дана інформація подається менеджерам.

4.3.2.16 Завдання для варіанта № 16.

Розробити програмне забезпечення керування банківськими рахунками. Користувачі мають доступ до власних рахунків та можуть вносити гроші на рахунок, витрачати їх або направляти на рахунок іншого клієнта банку. Адміністратор може змінювати стан проведених операцій, додавати нових користувачів до системи, визначати стан їх рахунку, змінювати їх кредитний ліміт. Система повинна надавати користувачу інформацію щодо стану його рахунку, всіх проведених операцій.

4.3.2.17 Завдання для варіанта № 17.

Розробити програмне забезпечення бронювання квитків на потяг. Адміністратор визначає доступні потяги, зазначаючи шлях їх слідування. Користувачі можуть бронювати квитки на потяг, зазначаючи його номер або виконуючи пошук. Система повинна забезпечувати можливість пошуку потягів, за допомогою яких можна дістатися з першої станції у даний день на другу станцію, та потягів, які дозволять дістатися на вказану станцію раніше вказаної дати та часу зі вказаної станції. Адміністратор повинен мати можливість переглядати кількість вільних місць на потяги, а користувач – переглядати власні бронювання, а також відмовлятися від них.

4.3.2.18 Завдання для варіанта № 18.

Розробити програмне забезпечення електронного замовлення книг у бібліотеці. Система повинна дозволяти читачам замовляти книги. Книги характеризуються автором, назвою, роком видання, видавництвом, кількістю сторінок. Бібліотекар вводить в систему інформацію про наявні книги та їх кількість. Система резервує книгу за користувачем, якщо хоча б один її екземпляр не зарезервований іншими читачами. Якщо вільних книг немає, то створюється черга. Якщо читач тримає книгу довше двох тижнів, то нараховується пеня. Читач не може резервувати книги, доки пеня не сплачена (визначається бібліотекарем).

4.3.2.19 Завдання для варіанта № 19.

Розробити програмне забезпечення продажу квитків на літак. Адміністратор визначає маршрути, дати перельотів, наявні місця. Користувачі мають можливість купувати квитки. При цьому можна або обрати конкретне місце в літаку, або за день до вильоту система автоматично має розподілити місця між всіма пасажирами, що не обрали місце. Спочатку виконується бронювання квитка, дійсне протягом доби. Якщо гроші за квиток внесено, то він вважається придбаним. Факт сплати за квиток можна моделювати випадковим чином. Адміністратору доступна статистика щодо продажів. Без реєстрації користувач не може купити квиток.

4.3.2.20 Завдання для варіанта № 20.

Розробити програмне забезпечення музичної бібліотеки. Кожна композиція характеризується назвою, виконавцем, жанром і альбомом, до якого вона належить (якщо визначено). Користувачі можуть прослуховувати композиції, виставляти їм або альбомам оцінки. Статистика власних прослуховувань та оцінок доступна кожному користувачу, а загальна статистика доступна всім користувачам. Статистика надається в розрізі жанрів, альбомів та виконавців.

4.3.2.21 Завдання для варіанта № 21.

Розробити програмне забезпечення проведення опитувань. Кожне опитування триває до того моменту, як його завершить адміністратор. Адміністратор створює нові опитування, змінює існуючі, передивляється результати опитування (в тому числі до завершення). Користувачам доступний перелік опитувань, і вони можуть проголосувати тільки один раз. Результат опитування

доступний користувачам тільки після його завершення. Користувачі можуть також залишати коментарі до опитувань.

4.3.2.22 Завдання для варіанта № 22.

Розробити програмне забезпечення електронного голосування. Система надає користувачам можливість зареєструватися. При цьому користувач не вважається зареєстрованим, доки адміністратор не підтвердить його реєстрацію. Адміністратор визначає перелік партій-кандидатів. Голосування відбувається у визначений адміністратором час. Протягом даного періоду користувачі мають можливість проголосувати за одного з кандидатів. Система підводить підсумки голосування після закінчення відведеного періоду. Результати визначаються за пропорційною системою. Партії-кандидати, що набрали частку голосів меншу за встановлену, участі у представницькому органі не беруть. Враховувати можливість роботи системи над декількома голосуваннями протягом її життєвого циклу.

4.3.2.23 Завдання для варіанта № 23.

Розробити програмне забезпечення підтримки роботи служби доставки. Модератори можуть створювати нові замовлення, надаючи їм номер та вносячи опис (маршрут, вага, інші параметри). Користувачі за номером можуть відслідковувати стан. Після того, як модератор зафіксував результат доставки, користувач може залишити відгук. Користувач може ставити питання модератору за замовленням, а модератор відповідати на них.

4.3.2.24 Завдання для варіанта № 24.

Розробити програмне забезпечення керування власним бюджетом. Користувачі можуть формувати перелік власних витрат, визначаючи для них назву, категорію, вартість. Система надає можливість аналізувати відсотковий розподіл за витратами, визначати найбільші з них, підраховувати загальні витрати за категоріями та загалом за вказаний період часу. Користувач може надавати доступ до перегляду власного бюджету деяким користувачам.

4.3.2.25 Завдання для варіанта № 25.

Розробити програмне забезпечення оренди автомобілів. Адміністратори визначають доступні автомобілі. Користувачі оформлюють бронювання, визначаючи дати, протягом яких вони будуть використовувати автомобіль. Система відслідковує, коли автомобіль вільний для використання. Користувачі можуть

скасовувати бронювання, за що нараховується пеня. У випадку використання нараховується сума до сплати.

4.3.2.26 Завдання для варіанта № 26.

Розробити програмне забезпечення продажу речей, що були у використанні. Користувачі визначають речі, їх опис та вартість, за якою вони б хотіли їх продати. Кожен інший користувач може сформувати заявку на купівлю даного товару, встановлюючи пропоновану ціну. Власник речі може завершити торгівлю власноруч. Інакше заявка закривається, якщо протягом доби ніхто не подав іншої пропозиції. Товар продається покупцю з найвищою ціною. Адміністратор може переглядати статистику за вирученими коштами за вказаний період (день, тиждень, місяць, квартал, рік).

4.3.2.27 Завдання для варіанта № 27.

Розробити програмне забезпечення керування банківськими кредитами. Користувачі мають банківський рахунок, на який можуть вносити гроші, а також брати певну суму в кредит. Максимальна сума кредиту визначається працівником банка для кожного користувача. Система розраховує плату за кредитом та знімає її за кожен місяць. Має враховуватись, що користувач може сплачувати за кредит наперед, тоді відсотки зменшуються. Якщо сума на рахунку від'ємна, то нараховуються додаткові відсотки. Адміністратор може отримувати статистичну інформацію про стан рахунків.

4.3.2.28 Завдання для варіанта № 28.

Розробити програмне забезпечення звітування за виконанням проєктів. Адміністратори вносять, редагують та вилучають дані про проєкти. Кожен проєкт характеризується описовими та числовими параметрами. Числові параметри визначають планові показники. Користувачі прив'язані до власних проєктів і не можуть переглядати ніякі інші. За власними проєктами вони можуть вносити фактичні показники. Експерти можуть переглядати всі проєкти, їх фактичні і планові показники та виставляти оцінки проєктам. Система підводить підсумки (середні, максимальні та мінімальні бали) за проєктами.

4.3.2.29 Завдання для варіанта № 29.

Розробити програмне забезпечення обміну короткими повідомленнями. Кожен користувач може надсилати повідомлення іншим користувачам, переглядати надіслані йому, при чому як загальний набір, так і окремо за обраним користувачем, з яким він має

діалог. Система повинна дозволяти адміністратору переглядати інформацію про кількість повідомлень, відправлену в системі за кожний день у заданому інтервалі, а також про кількість повідомлень, відправлену кожним користувачем.

4.3.2.30 Завдання для варіанта № 30.

Розробити програмне забезпечення відслідковування помилок. Керівники проєктів вносять інформацію про проєкти та визначають учасників проєктів. Учасники проєктів можуть вносити інформацію про виявлені помилки, а також змінювати стан на вирішений, додаючи коментарі про спосіб розв'язання. Керівники проєктів можуть переглядати дані про помилки за кожним проєктом, за всіма проєктами загалом, про всі вирішені або невирішені помилки.

4.3.3. Визначити проблеми, які було ідентифіковано в процесі розроблення програмного забезпечення і такі, що можуть реалізуватись під час подальших етапів життєвого циклу програмного забезпечення, та відповідні їм засоби управління.

4.3.4. Оформити звіт.

4.3.5. Відповісти на контрольні запитання.

4.4. Зміст звіту

4.4.1. Мета роботи.

4.4.2. Завдання на роботу.

4.4.3. Архітектура програмного забезпечення (з виділенням компонентів програмної системи та обґрунтуванням їх відповідного складу).

4.4.4. Текст розробленого програмного забезпечення.

4.4.5. Опис проблем та засобів управління ними.

4.4.6. Результати тестування: вхідні дані та результати роботи програми.

4.4.7. Висновки, що містять відповіді на контрольні запитання, а також відображають результати виконання роботи, їх критичний аналіз та аналіз досліджених засобів компоненто-орієнтованого програмування.

4.5. Контрольні запитання

4.5.1 Що таке компонент?

4.5.2 З чого складається архітектура програмного забезпечення, розробленого на основі компонентів?

4.5.3 Які технології розроблення компонентного програмного забезпечення існують?

4.5.4 Що таке фреймворк .NET?

4.5.5 Для чого необхідні програмні інтерфейси?

4.5.6 З яких етапів складається розроблення програмного забезпечення на основі компонентів мовою програмування C# на платформі .NET?

4.5.7 Яким чином застосування компонентно-орієнтованої парадигми програмування впливає на ризики розроблення програмного забезпечення?

4.5.8 Яким чином компоненти .NET розташовуються в пам’яті?

4.5.9 Що таке та яким чином працює CLR?

4.5.10 У чому полягає модель безпеки .NET?

4.5.11 Для чого призначені діаграми компонентів?

4.5.12 Які елементи можуть розташовуватися на діаграмах компонентів?

ЛІТЕРАТУРА

1. Навчальний посібник з дисципліни «Системний аналіз» для здобувачів спеціальності 122 – комп'ютерні науки [Текст] / В.М. Тонконогий, В.О. Вайсман, Л.В. Бовнегра, К.Г. Кіркопуло. – Одеса: Нац. ун-т «Одеська політехніка», 2022. – 84 с.

2. Дубровін, В.І. Прийняття рішень у процесі управління ризиками проєктів [Текст]: навчальний посібник / В.І. Дубровін, В.М. Льовкін. – Запоріжжя: ЗНТУ, 2012. – 196 с.

3. Панкратова, Н.Д. Системний аналіз [Текст]: теорія та застосування: підручник / Н.Д. Панкратова. – К.: Вид-во «Наукова думка» НАН України, 2019. – 352 с

4. Прокопенко, Т.О. Теорія систем і системний аналіз [Електронний ресурс]: навчальний посібник / Т.О. Прокопенко. – Черкаси: ЧДТУ, 2019. – 139 с.

5. Lutz, M. Learning Python [Текст] / M. Lutz ; 5th Edition. – O'Reilly Media, 2013. – 1643 p.

6. Коноваленко, І.В. Платформа .NET та мова програмування C# 8.0 [Текст]: навчальний посібник / І.В. Коноваленко, П.О. Марущак. – Тернопіль: ФОП Паляниця В. А., 2020. – 320 с.

7. Васильєв, О. Програмування мовою Python [Текст] / О. Васильєв. – Тернопіль: "Навчальна книга – Богдан", 2019. – 504 с.

8. Висоцька, В.А. Python [Текст]: алгоритмізація та програмування / В.А. Висоцька, О.В. Оборська. – Львів: Видавництво «Новий Світ – 2000», 2021. – 514 с.

9. Заяць, В.М. Логічне і функціональне програмування. Системний підхід. Підручник. – 2-ге видання, випр. та доповн. [Текст] / В.М. Заяць, М.М. Заяць. – Рівне: НУВГП, 2018. – 422 с.

10. Шушура, О.М. Системний аналіз [Текст] : навчальний посібник / О.М. Шушура, Н.К. Шатохіна. – К.: Редакційно-видавничий центр Державного університету телекомунікацій, 2019. – 63 с.

11. Python 3.11.3 documentation [Електронний ресурс]. – Режим доступу: https://docs.python.org/3/

12. Pygame Front Page – pygame v2.4.0 documentation [Електронний ресурс]. – Режим доступу: https://www.pygame.org/docs/

13. C# docs – get started, tutorials, reference. Microsoft Learn [Електронний ресурс]. – Режим доступу: https://learn.microsoft.com/uk-ua/dotnet/csharp/

14. Vu, H. Component Oriented Programming [Текст] : Object-Oriented and Beyond / Hieu Vu. – LAP LAMBERT Academic Publishing, 2017. – 188 p.