30. Тестування. Розповсюдження власних застосувань

Т30.1 Описати функцію, що повертає суму всіх доданків при заданому значенні *x*, що за абсолютною величиною не перевищують заданого ε > 0.
Скласти програму для тестування цієї функції при декількох значеннях x та ε:

а) $y = \ln(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} - \ldots (|x| < 1);$

б) $y = \frac{1}{1+x} = 1 - x + x^2 - x^3 + \ldots (|x| < 1);$

в) $y = \ln\frac{1+x}{1-x} = 2 \cdot \left[\frac{x}{1} + \frac{x^3}{3} + \frac{x^5}{5} + \ldots\right] \quad (|x| < 1);$

г) $y = \frac{1}{(1+x)^2} = 1 - 2 \cdot x + 3 \cdot x^2 - \ldots (|x| < 1);$

д) $y = \frac{1}{(1+x)^3} = 1 - \frac{2 \cdot 3}{2} \cdot x + \frac{3 \cdot 4}{2} \cdot x^2 - \frac{4 \cdot 5}{2} \cdot x^3 + \ldots (|x| < 1);$

е) $y = \frac{1}{1+x^2} = 1 - x^2 + x^4 - x^6 + \ldots (|x| < 1);$

є) $y = \sqrt{1+x} = 1 + \frac{1}{2} \cdot x - \frac{1}{2 \cdot 4} \cdot x^2 + \frac{1 \cdot 3}{2 \cdot 4 \cdot 6} \cdot x^3 - \ldots (|x| < 1);$

ж) $y = \frac{1}{\sqrt{1+x}} = 1 - \frac{1}{2} \cdot x + \frac{1 \cdot 3}{2 \cdot 4} \cdot x^2 - \frac{1 \cdot 3 \cdot 5}{2 \cdot 4 \cdot 6} \cdot x^3 + \ldots (|x| < 1);$

з) $y = \arcsin x = x + \frac{1}{2} \cdot \frac{x^3}{3} + \frac{1 \cdot 3}{2 \cdot 4} \cdot \frac{x^5}{5} + \ldots (|x| < 1).$

Т30.2 Описати функцію, що повертає
а) суму елементів матриці
б) мінімальний елемент матриці
в) максимальний елемент матриці
Скласти програму для тестування цієї функції.

Т30.3 Дано два рядки однакової довжини: s1 та s2. Відстанню Хемінга між s1 та s2 називають кількість позицій, у яких s1[i] != s2[i]. Описати функцію для обчислення відстані Хемінга двох рядків.
Скласти програму для тестування цієї функції.

Т30.4 Описати функцію для обчислення значення натурального числа за заданим рядком символів, який є записом цього числа у системі числення за основою b($2<b<16$). Використати функцію, яка за заданим символом повертає відповідну цифру у системі числення за основою b. Використати у цій функції твердження про стан програми assert для перевірки того, що відповідний символ є цифрою у системі числення за основою b. Обробити у підпрограмі помилку неправильного символу рядка та показати змістовне повідомлення про помилку.
Скласти програму для тестування цих функцій з урахуванням перевірки обробки виключення при неправильному символі рядка.

Т30.5 У текстовому файлі записана непорожня послідовність дійсних чисел, які розділяються пропусками в одному рядку та можуть бути розташовані у різних рядках. Визначити функцію обчислення найбільшого з цих чисел.
Забезпечити обробку помилок, якщо у файлі зустрічаються не дійсні числа.
Скласти програму для тестування цієї функції з урахуванням перевірки обробки виключення, якщо зустрілось не дійсне число.
а) Додатково підготувати тестове оточення (файл)
б) Додатково використати удавані об’єкти та функцію patch для заміни дій над файлом.

Т30.6 Клас Стек описано наступним чином:

```
class Stack:
    '''Реалізує стек на базі списку.
    '''
    def __init__(self):
        '''Створити порожній стек.
        '''
        self._lst = []          #список елементів стеку

    def isempty(self):
        '''Чи порожній стек?.
        '''
        return len(self._lst) == 0

    def push(self, data):
        '''Вштовхнути елемент у стек.
        '''
        self._lst.append(data)

    def pop(self):
        '''Взяти елемент зі стеку.
        '''
        if self.isempty():
            print('Pop: Стек порожній')
            exit(1)
        data = self._lst.pop()
```

```
        return data
```

Описати власний клас на базі Stack, у якому перевизначити метод pop так, щоб він ініціював виключення при спробі взяти елемент порожнього стеку. Скласти програму для тестування цього класу (усіх його методів) з урахуванням перевірки обробки виключення.

Т30.7 Клас Черга описано наступним чином:

```
class Queue:
    '''Реалізує чергу на базі списку.
    '''
    def __init__(self):
        '''Створити порожню чергу.
        '''
        self._lst = []                    #список елементів черги

    def isempty(self):
        '''Чи порожня черга?.
        '''
        return len(self._lst) == 0

    def add(self, data):
        '''Додати елемент в кінець черги.
        '''
        self._lst.append(data)

    def take(self):
        '''Взяти елемент з початку черги.
        '''
        if self.isempty():
            print('Take: Черга порожня')
            exit(1)
        data = self._lst.pop(0)          #перший елемент черги - це нульовий
елемент списку
        return data

    def __del__(self):
        '''Закінчити роботу з чергою.
        '''
        print('Deleting queue')
        del self._lst
```

Описати власний клас на базі Queue, у якому перевизначити метод take так, щоб він ініціював виключення при спробі взяти елемент порожньої черги. Скласти програму для тестування цього класу (усіх його методів) з урахуванням перевірки обробки виключення.

Т30.8 Клас Кільцевий список описано наступним чином:

```
class Rlist:
    '''Реалізує кільцевий список на базі списку.
    '''
    def __init__(self):
        '''Створити порожній список.
        '''
        self._lst = []                   #список елементів
        self._cur = None                 #індекс поточного елемента

    def len(self):
        '''Довжина списку.
        '''
        return len(self._lst)
```

```
    def next(self):
        '''Перейти до наступного елемента.
        '''
        l = self.len()
        if l != 0:
            if self._cur == l-1:    #для (l-1) елемента наступним буде
нульовий
                self._cur = 0
            else:
                self._cur += 1

    def getcurrent(self):
        '''Повернути поточний елемент.
        '''
        if self.len() == 0:
            print('getcurrent: список порожній')
            exit(1)
        data = self._lst[self._cur]
        return data

    def update(self, data):
        '''Оновити поточний елемент.
        '''
        if self.len() == 0:
            print('update: список порожній')
            exit(1)
        self._lst[self._cur] = data

    def insert(self, data):
        '''Вставити елемент перед поточним.
        '''
        if self.len() == 0:             #якщо список порожній
            self._lst.append(data)      #додаємо елемент, він стає поточним
            self._cur = 0
        else:
            self._lst.insert(self._cur,data) #інакше вставляємо елемент перед
поточним
            self._cur += 1                   #щоб поточний елемент не
змінився, треба індекс збільшити на 1

    def delete(self):
        '''Видалити поточний елемент.
        '''
        if self.len() == 0:
            print('delete: список порожній')
            exit(1)
        l = self.len()
        del self._lst[self._cur]
        if l == 1:                  #якщо список після видалення елемента
спорожніє
            self._cur = None
        elif self._cur == l-1:      #якщо поточним був останній елемент
списку
            self._cur = 0           #поточним стане елемент з індексом 0
        #else: pass                  якщо поточним був не останній елемент,
нічого не робити

    def __del__(self):
        '''Закінчити роботу зі списком.
        '''
        del self._lst
```

Описати власний клас на базі Rlist, у якому перевизначити методи getcurrent, update, delete так, щоб вони ініціювали виключення у випадку порожнього списку.
Скласти програму для тестування цього класу (усіх його методів) з урахуванням перевірки обробки виключення.

Т30.9 Описати клас для роботи з відрізками на числовій осі. Для відрізку передбачити поля:

(*a*, *b*, *empty)*

де *a*, *b* - границі відрізку, *empty* - ознака того, що відрізок порожній.
Реалізувати методи:

1) зробити відрізок *t* порожнім;
2) чи порожній відрізок *t*;
3) покласти відрізок *t* рівним *a*, *b*;
4) покласти відрізок *t* рівним перетину відрізків *t1*, *t2*.

Скласти програму для тестування цього класу (усіх його методів).

Т30.10 Описати клас для реалізації мультимножини цілих чисел на базі словника. Мультимножина - це множина в якій для кожного елемента запам'ятовується не лише його входження, але й кількість входжень.

Кількість входжень елемента *k* (0<=*k*<=*n*) у мультимножину - це значення елемента словника з ключем *k*.
Реалізувати методи:

1) зробити мультимножину порожньою;
2) чи є мультимножина порожньою;
3) додати елемент до мультимножини;
4) забрати елемент з мультимножини (кількість входжень елемента зменшується на 1, якщо елемент не входить - відмова);
5) кількість входжень елемента у мультимножину;
6) об'єднання двох мультимножин (в результаті об'єднання кідькість входжень елемента визначається як максимальна з двох мультимножин);
7) перетин двох мультимножин (в результаті кількість входжень елемента визначається як мінімальна з двох мультимножин);

Скласти програму для тестування цього класу (усіх його методів).

Т30.11 У текстовому файлі є дати, задані у форматі dd.mm.yyyy або у форматі yyyy/mm/dd. Також день та/або місяць може містити одну цифру, а не 2. Привести всі дати до єдиного формату dd.mm.yyyy.
Вказівка: використати регулярні вирази та функцію (метод) sub.
Скласти програму для тестування цієї програми.
а) Додатково підготувати тестове оточення (файл)
б) Додатково використати удавані об'єкти та функцію patch для заміни дій над файлом.

в) Додатково використати удавані об’єкти та функцію patch для заміни дій над регулярними виразами.

Т30.12 За допомогою регулярних виразів розбити текст у текстовому файлі на речення.
Скласти програму для тестування цієї програми.
а) Додатково підготувати тестове оточення (файл)
б) Додатково використати удавані об’єкти та функцію patch для заміни дій над файлом.
в) Додатково використати удавані об’єкти та функцію patch для заміни дій над регулярними виразами.

Т30.13 У текстовому файлі містяться дати у форматі dd.mm.yyyy або підкреслення для запису дат вручну __.__.____ Знайти всі дати у тексті. Замість підкреслень вставити поточну дату. Зберегти оновлений текст.
Скласти програму для тестування цієї програми.
а) Додатково підготувати тестове оточення (файл)
б) Додатково використати удавані об’єкти та функцію patch для заміни дій над файлом.
в) Додатково використати удавані об’єкти та функцію patch для заміни дій над регулярними виразами.

Т30.14 Скласти програму порівняння двох каталогів. Програма повинна показувати різницю у каталогах, тобто файли, що присутні в одному каталозі та відсутні у іншому. Якщо файл присутній у одному та іншому каталозі, він не повинен бути показаний. Результат роботи програми слід спрямувати у текстовий файл.
Скласти програму для тестування цієї програми.
а) Додатково підготувати тестове оточення (файл)
б) Додатково використати удавані об’єкти та функцію patch для заміни дій над файлом.
в) Додатково використати удавані об’єкти та функцію patch для заміни дій над каталогами (модуль os).

Т30.15 Скласти програму порівняння файлів з однаковим ім’ям у двох каталогах. Для таких пар файлів треба показати різницю у часі створення файлу, тобто, який з двох файлів цієї пари був створений пізніше. Якщо файли були створені одночасно, то нічого показувати не потрібно. Результат роботи програми слід спрямувати у текстовий файл.
Скласти програму для тестування цієї програми.
а) Додатково підготувати тестове оточення (файл)
б) Додатково використати удавані об’єкти та функцію patch для заміни дій над файлом.
в) Додатково використати удавані об’єкти та функцію patch для заміни дій над каталогами (модуль os).

Т30.16 Деяка існуюча програма записує у системний журнал. Системні журнали містяться у заданому каталозі та створюються по декілька файлів на кожну дату. Відповідно, дата та час створення є частиною імені файлу.
Треба скласти програму, яка архівує системні журнали у заданому каталозі за попередні дати у форматі tar.gz.
Скласти програму для тестування цієї програми.
а) Додатково підготувати тестове оточення (файл)
б) Додатково використати удавані об'єкти та функцію patch для заміни дій над файлом.
в) Додатково використати удавані об'єкти та функцію patch для заміни дій над каталогами (модуль os).

Т30.17 Скласти програму, яка читає прогноз погоди у заданому місті з сайту sinoptik.ua та зберігає у файлі Excel у окремому рядку поточну дату та прогнози максимальної та мінімальної температури на кожний з наступних 5 днів.
Запит на погоду у заданому місті:
https://sinoptik.ua/погода-<місто>
Наприклад,
https://sinoptik.ua/погода-киев
Скласти програму для тестування цієї програми.
а) Додатково використати удавані об'єкти та функцію patch для заміни дій з читання даних з мережі.
б) Додатково використати удавані об'єкти та функцію patch для заміни дій над файлами MS Excel.

Т30.18 Скласти програму, яка читає точний час з сайту https://time.online.ua/in/kyiv/, перевіряє, чи відповідає час на локальному комп'ютері точному часу. Якщо різниця становить більше 1 хвилини, то виправляє час на локальному комп'ютері.
Час можна дізнатись у тегу <script>:

```
<div class="city-clock-time">
    <div class="clock-img" id="clock_id"></div>
    <script>
    draw_clock('clock_id',2017,02,23,23,50,09);
    setInterval("update_clocks()",1000);
    </script>
</div>
```

Підказка: Для встановлення часу у linux виконати команду операційної системи 'date -s "2 OCT 2010 18:00:00"'; у Windows виконати команду операційної системи 'time 18:00:00' (замість 18:00:00 використати потрібний час).
Скласти програму для тестування цієї програми.
а) Додатково використати удавані об'єкти та функцію patch для заміни дій з читання даних з мережі.

б) Додатково використати удавані об’єкти та функцію patch для заміни дій зі встановлення дати у певній ОС.

Т30.19 Скласти програму, яка працює в оточенні веб-сервера, для введення рядка та повернення всіх різних слів цього рядка. Слова повертати у форматі JSON. Структуру даних JSON визначити самостійно.
Скласти програму для тестування цієї програми. Додатково використати удавані об’єкти та функцію patch для заміни дій з читання даних з мережі та повернення відповіді клієнту.

Т30.20 Скласти програму, яка працює в оточенні веб-сервера, для розв’язання задачі.
Знайти у даному рядку символ та довжину найдовшої послідовності однакових символів, що йдуть підряд.
Повернути символ та довжину найдовшої послідовності символів у форматі XML. Структуру даних XML визначити самостійно.
Скласти програму для тестування цієї програми. Додатково використати удавані об’єкти та функцію patch для заміни дій з читання даних з мережі та повернення відповіді клієнту.

Т30.21 Скласти програму для роботи з базою даних, що містить логіни та паролі входу до систем. Для кожної системи у БД зберігається неформальна назва, адреса (якщо є), логін та пароль. Реалізувати функції додавання системи та повернення адреси, логіну та паролю за назвою системи.
Скласти програму для тестування цієї програми.
а) Додатково підготувати тестове оточення (базу даних)
б) Додатково використати удавані об’єкти та функцію patch для заміни дій над базою даних.

Т30.22 Скласти програму для роботи з базою даних, що містить означення понять та їх опис. Реалізувати функції додавання поняття та повернення опису за введеним поняттям.
Скласти програму для тестування цієї програми.
а) Додатково підготувати тестове оточення (базу даних)
б) Додатково використати удавані об’єкти та функцію patch для заміни дій над базою даних.

Т30.23 Створити структуру каталогів та описати необхідні файли для розповсюдження вибраного власноруч написаного модуля. Підготувати заархівований файл та встановити його у Python з диску.