Лабораторна робота №2

Реалізувати програму з графічним інтерфейсом користувача. Для реалізації інтерфейсу можна використовувати Qt або інший фреймворк (за бажанням студента, погодивши з викладачем).

Далі наведені можливі програми для реалізації, для кожної з яких є основні можливості (які треба обов’язково реалізувати) та список додаткових можливостей (з них обов’язково треба обрати хоча б декілька для реалізації, хоча за бажання можна реалізувати й усі, отримавши за це більше балів). Варіантом цієї лабораторної роботи є номер програми та номери додаткових можливостей, які будуть реалізовані.

За бажанням можна запропонувати свій власний варіант – яку програму хочеться реалізувати. Складність програми має приблизно відповідати вже наведеним варіантам. Запропонований варіант треба узгодити з викладачем. Також треба навести короткий опис реалізованої програми.

***Програма 1 – розумний таймер***

*Базові можливості:*

* Можливість запуску таймерів, звуковий та візуальний сигнал по завершенню часу.
* Можливість виконання певних налаштовуваних дій по завершенню часу (наприклад, запуск програми чи відкриття документу).
* Таймери на певний проміжок часу (власне таймери) або до заданого моменту часу (будильники) – бажано реалізувати як один список, можливо з додатковою фільтрацією
* Можливість одночасного запуску декількох (довільної кількості) таймерів,
* Зручний інтерфейс для перегляду списку таймерів, керування таймерами

*Додаткові можливості:*

1. Фільтри для вибору таймерів за різними параметрами.
2. Таймери різних типів – дозволяють налаштовувати сигнал, значення часу, інші параметри.
3. Різні формати задання часу таймерів – час від даного моменту, час спрацювання, різні формати часу, інші можливості; зручне та швидке введення часу
4. Підказки під час запуску таймерів – можливість задати певні залежності між таймерами, наприклад, після спрацювання таймера типу 1 рекомендовано запустити таймер типу 2
5. Завчасно налаштовані групи таймерів з можливість швидкого запуску всіх таймерів групи
6. Можливість режиму “Do not disturb” – певного часу, коли сигнали таймерів не спрацьовують.
7. Можливість запам’ятовувати запущені таймери і продовжувати роботу після перезапуску застосунку чи ОС.
8. Підтримка різних часових зон.
9. Додатково до переліку таймерів – який таймер спрацює найпершим.
10. В панелі задач (task bar) індикація прогресу до найближчого таймера.
11. В панелі задач (task bar) кількість активних таймерів.
12. Можливість впорядкування переліку таймерів: сортування за різними критеріями, зміна порядку «вручну»

***Програма 2 – offline file manager***

*Базові можливості:*

* Можливість зберігання інформації про файли та каталоги на зовнішніх носіях (флеш-диски, зовнішні диски, CD/DVD) – інформація має залишатись доступною навіть після відключення носія (в read-only режимі)
* Зберігання інформації після перезавантаження програми
* Можливість додавати користувацькі метадані до файлів/каталогів
* Якщо носій підключено – можливість відкрити системний переглядач файлів для заданого файлу чи каталогу.

*Додаткові можливості:*

1. Можливість створення віртуальних каталогів, які об’єднують дані з різних джерел
2. Пошук за метаданими (ім’я файлу/каталогу, тип, час створення/редагування, хеш/контрольна сума, користувацькі метадані)
3. Інформація про кількість точних копій файлу чи каталогу на відомих носіях
4. Генерація і запуск команд для копіювання чи синхронізації даних між різними носіями (наприклад, з використанням rsync, robocopy чи інших подібних засобів)
5. Підтримка хмарних сховищ (наприклад, Google Drive, Dropbox, …)
6. Підтримка даних на FTP серверах
7. Підтримка серверних git-репозиторіїв
8. Можливість обмеження глибини перегляду структури каталогів на носії (як під час збереження інформації, так і під час її перегляду користувачем)
9. Можливість додавання фільтрів, що обмежують перегляд структури каталогів

***Програма 3 – демонстрація роботи алгоритмів чи структур даних***

*Базові можливості:*

* Графічна ілюстрація певної структури даних (де зберігаються дані, куди показують вказівники, …)
* Покрокова ілюстрація виконання алгоритму – як змінюється стан структури даних та додаткових змінних на кожному кроці алгоритму
* Можливість рухатись як вперед, так і назад по крокам алгоритму
* Список підтримуваних алгоритмів/структур даних, можливість вибору зі списку для перегляду демонстрації.

*Додаткові можливості:*

1. Можливість користувачу задавати параметри структури даних – наприклад, кількість елементів, значення елементів
2. Можливість збереження та відновлення проміжного стану демонстрації.
3. Можливість вибору якихось елементів демонстрації для відслідковування – вони виділяються, наприклад кольором чи шрифтом, відображається інформація про їх зміни.
4. Можливість підключення нових демонстрацій без перекомпіляції основної програми.

***Програма 4 – демонстрація випадкових подій***

*Базові можливості:*

* Можливість задати кілька випадкових подій – задаються можливі результати та ймовірності кожного результату.
* Запуск моделювання подій заданої кількості разів
* Список результатів моделювання – які події стались на кожному кроці.

*Додаткові можливості:*

1. Підтримка інших видів випадкових подій – вибір елементів зі списку без повернення («колода карт»), випадкові блукання по прямій/площині/графу, випадкові точки в множинах на площині, … (як джерело інших ідей щодо можливих типів подій, можна подивитись <https://www.random.org/> )
2. Збір певної статистики – обчислення середніх значень, кількості подій з певної категорії, …
3. Можливість задати та зберегти seed value генератора випадкових чисел, щоб відтворити ту саму ситуацію.
4. Можливість підключення нових демонстрацій без перекомпіляції основної програми.

***Програма 5 – перевірка завдань з програмування***

*Базові можливості:*

* Можливість задавати кілька джерел коду – репозиторії, онлайн файлові сховища, локальна файлова система.
* Прив’язка коду до студентів.
* Можливість додавати коментарі, прив’язані до конкретного коду
* Генерація звітів з підсумками коментарів

*Додаткові можливості:*

1. Копіювання коду для запуску у спеціально налаштовані середовища
2. Перевірка виконання тестів
3. Перевірка на заданих вхідних даних
4. Запуск статичних аналізаторів коду і збір інформації
5. Виміри продуктивності коду – час виконання, використання пам’яті, …
6. Збір статистики з репозиторію – кількість комітів, кількість гілок, активність за певний період часу
7. Підтримка версіонування для коду, що зберігається не в репозиторіях source control
8. Можливість пошуку спеціальних зразків неправильного коду (налаштовуються користувачем)
9. Можливість створення скриптів для опису додаткових перевірок
10. Можливість створення плагінів для розширення функціональності

***Програма 6 – колекція посилань на ресурси (книги, веб-ресурси, аудіо та відео матеріали, …)***

*Базові можливості:*

* Можливість вводити посилання різних типів.
* Прив’язування посилань до певних контекстів (проектів, каталогів, …)
* Можливість перегляду посилань.
* Пошук посилань (за текстом)
* Збереження списків посилань у зовнішні документи.

*Додаткові можливості:*

1. Ієрархічні контексти
2. Одне посилання може належати до кількох контекстів
3. Зв’язки між посиланнями (наприклад, друкована книга та електронна версія, переклади на різні мови, аудіо/відео та текстова версія, …)
4. Пошук за різними критеріями
5. Пошук схожих посилань у зовнішніх пошукових системах
6. Додавання посилань з браузера (наприклад, зайшли на сторінку книги в Amazon, додали посилання на цю книгу)
7. Імпорт посилань з зовнішніх документів
8. Додавання коментарів до посилань
9. Генерація текстового подання посилань відповідно до різних стандартів (наприклад, ДСТУ ГОСТ 7.1:2006, ДСТУ 8302:2015, Harvard, BibTeX)