МІНІСТЕРСТВО НАУКИ І ОСВІТИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра

автоматизованих систем обробки інформації і управління

**КУРСОВА РОБОТА**

з дисципліни

“ Об’єктно-орієнтоване програмування ”

на тему

“ Векторний графічний редактор ”

Студента 2 курсу ІП-54 групи

напряму підготовки «Програмна інженерія»

спеціальності «Програмне забезпечення систем»

Литвинюк Д.Р.

Керівник Порєв В.М.

Національна оцінка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_\_\_\_Оцінка: ECTS \_\_\_\_\_

|  |  |  |  |
| --- | --- | --- | --- |
| Члени комісії |  |  |  |
|  | (підпис) |  | (вчене звання, науковий ступінь, прізвище та ініціали) |
|  |  |  |  |
|  | (підпис) |  | (вчене звання, науковий ступінь, прізвище та ініціали) |

Київ – 2016 рік

зміст

[Вступ 4](#_Toc469483215)

[1 постановка задачі 5](#_Toc469483216)

[2 Обгрунтування проектного рішення 6](#_Toc469483217)

[3 Аналіз можливих варіантів рішення завдання. 8](#_Toc469483218)

[4 теоретичні положення 9](#_Toc469483219)

[5 ОПИС РЕАЛІЗАЦІЇ ПРОЕКТУ 11](#_Toc469483220)

[Висновки 16](#_Toc469483221)

[Перелік Посилань 17](#_Toc469483222)

[Додаток А Технічне завдання 18](#_Toc469483223)

[Додаток Б Тексти програмного коду 21](#_Toc469483224)

Вступ

В курсовій роботі реалізується програма, призначена для створення векторних зобрражень.

Головна мета КР полягає у закріпленні, поглибленні та узагальненні базових теоретичних знань, отриманих під час вивчення дисципліни «Об’єктно-орієнтоване програмування», їх застосуванні до комплексного вирішення конкретного фахового завдання.

Основними цілями роботи є:

* надбання досвіду роботи з літературними та фондовими матеріалами, вміння узагальнювати та аналізувати наукову інформацію, виробляти власне ставлення до наукової чи практичної проблеми;
* набуття навичок використання основ алгоритмізації та програмування на алгоритмічних мовах високого рівня з використанням принципів об’єктно-орієнтованого програмування;
* набуття теоретичних знань та практичних навичок в області використання сучасних систем створення програмного забезпечення та освоєння принципів та методів сучасних технологій програмування;
* проведення ґрунтовного аналізу отриманих результатів і формування змістовних висновків стосовно їх якості.

У курсовій роботі можна створювати 7 фігур:

* лінію;
* штрих-пунктирну лінію;
* прямокутник;
* еліпс;
* куб;
* циліндр;
* ромб.

# постановка задачі

Користувач вводить зображення елеметів, які складають сцену. Елементами можуть бути : лінія, штрих-пунктирна лінія, прямокутник, еліпс, куб, циліндр, ромб. Програма повинна відповідати таким вимогам:

* + Введення кожного елементу виконується мишею після натискування відповідної кнопки на Toolbar.
  + Стрічка Toolbar розташовується у головному вікні під меню. Потрібно передбачити підказки (Tooltips) для кожної кнопки.
  + Потрібно передбачити скролінг по горизонталі та вертикалі, щоб забезпечити створення зображення розмірами 2000×2000. Передбачити коректне реагування на можливу зміну розмірів головного вікна.
  + Сцена може створюватися заново, або завантажуватися з текстового файлу.
  + Формат файлів сцен – текстовий.
  + Зміна кольору заливки та контуру.

# Обгрунтування проектного рішення

В даній програмі, я використовую такі класи:

* Клас, який є обробником основних повідомлень,  
  важливих для редагування: **ShapeObjectEditor**.
* Абстракний клас, від якого наслідуються усі класи графічні об’єкти: **Shape**.
* Класи, графічі об’єкти, які наслідуються від **Shape**: **EllipseShape, RectShape, RombShape, PuncktLineShape, CubeShape, LineShape, CilinderShape**.
* Абстракний клас, від якого наслідуються усі класи редагування об’єктів: **Editor**
* Базовий клас для забезпечення поліморфізму редагування об’єктів, який наслідується від Editor: ShapeEditor
* Класи редагування об’єктів, які наслідуються від ShapeEditor: EllipseEditor, RectEditor, RombEditor, PuncktLineEditor, CubeEditor, LineEditor, CilinderEditor.
* Клас, який відповідає за скроллінг: ScrollOperations.

Також я використовую модуль для панелі інструментів ToolBar.

На початку роботи программи ствоорюється об’єкт класу ShapeObjectEditor. Потім при виборі на панелі інструментів чи у меню певної фігури, створюється об’єкт класу ShapeEditor, усередені ShapeObjectEditor. Потім оброблюється повідомлення користовуча, які оброблює спочатку ShapeObjectEditor, а потім ShapeEditor. При натиснені лівої кнопки миші і до її відпущення малюється «гумовий слід» фігури, а потім створюється об’кт класу Shape і додається в кінець массиву графічних об’єктів. Згодом відбувається відображення оновленого масиву фігур. Також ShapeEditor та ShapeObjectEditor відповідають за запис у файл та читання з файлу ім’я яких передаються з обробника повідомлення,а також змінну кольорів, які теж передаються з обробника повідомленнь. Діалогові вікна для запису у файл, читанню з файлу та змінну кольорів, викликаються з обробника повідомленнь.

# Аналіз можливих варіантів рішення завдання.

Існує багато можливих варіантів рішення задачі створення векторного редактура на С++ з використанням Windows API. Знаходження оптимального рішення обов’зково спричинить палку суперечку між прогрмістами. Розглянемо деякі частини програми для яких очевидні декілька варіантів реалізації, які відрізняються від моєї:

* Можна було не створювати клас ShapeObjectEditor, а створювати об’єкт ShapeEditor всередині обробника повідомленнь, але це збільшило б розмір коду головного файлу програми і зробило б не зручним його читання.
* Можна було створити окрему функція для створення підказок(Tooltips).
* Можна було створити окремі класи для керування діалоговими вікнами запису у файл, читанням з файлу та змінною кольорів.
* Можна було не створювати клас, який відповідає за скроллінг, а реалізувати скроллінг усередині обробника повідомленнь.
* Можна було використати множине наслідування для створення куба, циліндра та ромба але це спричинило б проблем з їх заливкою.
* Можна було використати для збереження фігур вектор, а не масив.

# теоретичні положення

**Ве́кторна гра́фіка** — створення зображення в комп'ютерній графіці з сукупності геометричних примітивів — (точок, ліній, кривих, полігонів), тобто об'єктів, які можна описати математичними виразами.

**Лі́нія**— геометричний об'єкт, геометричне місце точок, що задовільняє певне рівняння.

**Прямоку́тник** — це чотирикутник, усі кути якого прямі[1]. Протилежні сторони прямокутника рівні. Є окремим випадком паралелограма[1].

**Еліпсом** називається множина всіх точок площини, для кожної з яких сума відстаней до двох даних точок F1 і F2 цієї площини є величина стала, більша за відстань між F1 і F2.

**Куб** або **гексаедр** — правильний многогранник, кожна грань якого є квадратом. Окремий випадок паралелепіпеда і призми.

**Цилі́ндр** — геометричне тіло, обмежене замкнутою циліндричною поверхнею і двома паралельними площинами, що перетинають її.

**Ромб** — це паралелограм, у якого всі сторони рівні.

**Windows Api** (application programming interfaces) — загальне найменування для цілого набору базових функцій інтерфейсів програмування застосунків операційних систем сімейства Windows корпорації Майкрософт. Є найпрямішим способом взаємодії застосунків з Windows. Для створення програм, що використовують Windows API, Майкрософт випускає SDK, який називається Platform SDK і містить документацію, набір бібліотек, утиліт і інших інструментальних засобів.

**Те́кстовий файл** — форма подання послідовності символів у комп'ютері, де кожен символ із задіяного набору символів кодується одним байтом чи послідовністю двох, трьох і т. д. байтів. На відміну від терміна «текстовий формат», що характеризує вміст даних, термін «текстовий файл» стосується файлу та характеризує його як контейнер, який зберігає такі дані.

**Діало́гове вікно́** — особливий тип вікна, яке задає запитання і дозволяє вибрати варіанти виконання дії, або ж інформує користувача. Діалогові вікна зазвичай відображаються тоді, коли програмі або операційній системі для подальшої роботи потрібна відповідь.

**Пане́ль інструме́нтів** (англ. toolbar) — елемент графічного інтерфейсу користувача, призначений для розміщення на ньому кількох інших елементів.

# ОПИС РЕАЛІЗАЦІЇ ПРОЕКТУ

Таблиця 5.1 – Опис класів

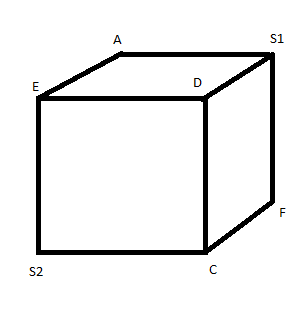
|  |  |
| --- | --- |
| Ім’я | Опис |
| **Клас ShapeObjectEditor** | |
| Поля | |
| ShapeEditor \*pse | Використовується для створення фігур |
| bool onLBdown = false | Кнопка натиснута |
| bool onLBup = false | Кнопка відпущена |
| Методи | |
| void StartRombEditor(COLORREF,COLORREF); | створює новий об'єкт RombEditor |
| void StartLineEditor(COLORREF, COLORREF); | створює новий об'єкт LineEditor |
| void StartRectEditor(COLORREF, COLORREF); | створює новий об'єкт RectEditor |
| Void StartEllipseEditor(COLORREF, COLORREF); | створює новий об'єкт EllipseEditor |
| void StartPuncktLineEditor(COLORREF,COLORREF); | створює новий об'єкт PuncktLineEditor |
| void StartCubeEditor(COLORREF, COLORREF) | створює новий об'єкт CubeEditor |
| void StartCilinderEditor(COLORREF, COLORREF) | створює новий об'єкт CilinderEditor |
| void StartWriteFile(TCHAR\*); | виконується запис у файл |
| void StartReadFile(TCHAR\*); | виконується зчитування з файлу |
| void Reset(HWND); | Перевантаження |
| void OnLBdown(HWND); | функція орбробки повідомлення натиснення лівої клавіши миші |
| void OnLBup(HWND, int xk, int yk); | функція орбробки повідомлення відпусксання лівої клавіши миші |
| void OnMosuseMove(HWND) | функція орбробки повідомлення руху миші |
| void OnPaint(HWND,HDC, int xk, int yk) | функція малювання фігур |
| void OninitMenuPopup(HWND, WPARAM) | позначає вибраний елемент у меню |
| void PressButton(HWND) | позначає вибраний елемент на панелі інструментів |
| void CreateNewScene() | створення нової сцени |
| **Клас Editor** | |
| Методи | |
| Virtual void OnLBdown(HWND) = 0; | функція орбробки повідомлення натиснення лівої клавіши миші |
| Virtual void OnLBup(HWND, int xk, int yk) = 0; | функція орбробки повідомлення відпусксання лівої клавіши миші |
| Virtual void OnMosuseMove(HWND) = 0 | функція орбробки повідомлення руху миші |
| Virtual void OnPaint(HWND,HDC, int xk, int yk) = 0 | функція малювання фігур |
| **Клас ShapeEditor** | |
| Поля | |
| static int curr\_length | поточна довжина масиву pcshape |
| static Shape \*pcshape[MY\_SHAPE\_ARRAY\_SIZE] | масив фігур |
| POINT pt\_old; | початкові координати |
| POINT pt\_start | кінцеві координати |
| COLORREF brColor | колір заливки |
| COLORREF penColor | колір контуру |
| Методи | |
| Void OverMeesage() | повідомлення про закінчення місця у масиві pcshape |
| Void CreateNewScene() | створення нової сцени |
| void StartWriteFile(TCHAR\*); | виконується запис у файл |
| void StartReadFile(TCHAR\*); | виконується зчитування з файлу |
| void OnLBdown(HWND); | функція орбробки повідомлення натиснення лівої клавіши миші |
| void OnLBup(HWND, int xk, int yk); | функція орбробки повідомлення відпусксання лівої клавіши миші |
| void OnMosuseMove(HWND) | функція орбробки повідомлення руху миші |
| void OnPaint(HWND,HDC, int xk, int yk) | функція малювання фігур |
| Virtual void OninitMenuPopup(HWND, WPARAM) | позначає вибраний елемент у меню |
| Virtual void PressButton(HWND) | позначає вибраний елемент на панелі інструментів |
| **Клас Shape** | |
| Поля | |
| long xs1, ys1, xs2, ys2; |  |
| std::string type; |  |
| COLORREF brColor | колір заливки |
| COLORREF penColor | колір контуру |
| Методи | |
| void Set(long x1,long y1, long x2, long y2,COLORREF,COLORREF,int xk,int yk); | ініціалізу об'єкт shape |
| virtual void Show(HDC,int xk, int yk) = 0; | малює об'єкт shape |
| std::string RetData(); | повертає дані про об'єкт shape |

Розглянемо, як створюються різні графічні об’єкти методом Show().

Для лінії та штрих-пунктирної лінії, використовуються стандартні фукції MoveToEx(початкова точка) LineTo(кінцева точка),

Для прямокутника та еліпса використовуються стандартні фукції Rectangle та Ellipse, які потребують лише координатии двох протилежних кутів.

Для малювання куба я використовую 3 полігони, для яких потрібно 7 точок. Маючи координати 2 точок S1 та S2 які задає користувач, я вираховую координати 5 інших.



xs\_D = xs\_1 - (xs\_1 - xs\_2) / 3,

ys\_D = ys\_1 + (ys\_2 - ys\_1) / 3,

xs\_A = xs\_2 + (xs\_1 - xs\_2) / 3,

ys\_A = ys\_1,

xs\_E = xs\_2,

ys\_E = ys\_D,

xs\_C = xs\_D,

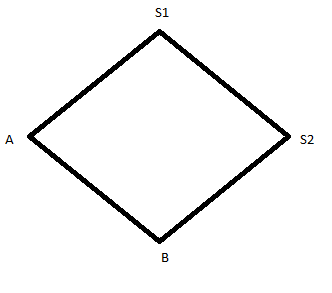
ys\_C = ys\_2,

xs\_F = xs\_1,

ys\_F = ys\_2 - (ys\_2 - ys\_1) / 3;

Відмальовуються 3 полігони: EA(S1)D, ED(S2)C, SC(S1)F.

Для малювання ромба я використовую 1 полігон, для цьгог маючи точки S1 та S2 прораховую координати 2 інших.



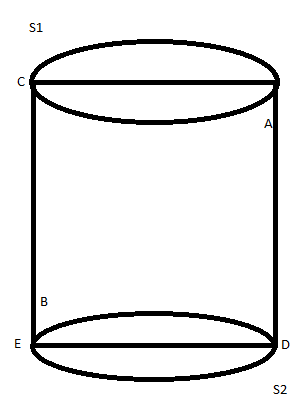
xs\_A = xs\_1 - (xs\_2 - xs\_1),

ys\_A = ys\_2,

xs\_B = xs\_1,

ys\_B = ys\_1 + ((ys\_2 - ys\_1)\* 2);

Для малювання циліндра я використовую 2 еліпси, прямокутник та лінію.



Спочатку малюється еліпс за кутами B(S2), потім прямокутник CD, на нього накладається еліпс AS1, а нижня сторона прямокутника замальовується прямою ED.

Висновки

В ході виконання курсової роботи був розроблений векторний графічний редактор. В якому можливо створювати зображення за допомогоя 7 фігур: лінії, штрих-пунктирною лінії, прямокутника, еліпса, куба, циліндра та ромба. Також можливо змінювати колір фігури, створення зображеня заново, або завантажувати його з текстового файлу та зберігати у файл. Реалізована панель інструментів та підказки для неї.

Була побудована діаграма класів до програми, та діаграма #include-ієрархії модулів.

Під час виконання роботи були набутті теоретичні та практичні навички в області використання сучасних систем створення програмного забезпечення та освоєння принципів та методів сучасних технологій програмування. Також було вироблене вміння застосовувати методи обчислювальної математики та прикладного програмування для розв’язання прикладних задач.

З використанням мови програмування С++ та Windows API графічної було розроблено програму, що реалізує векторний графічний редактор.

# Тексти програмного код