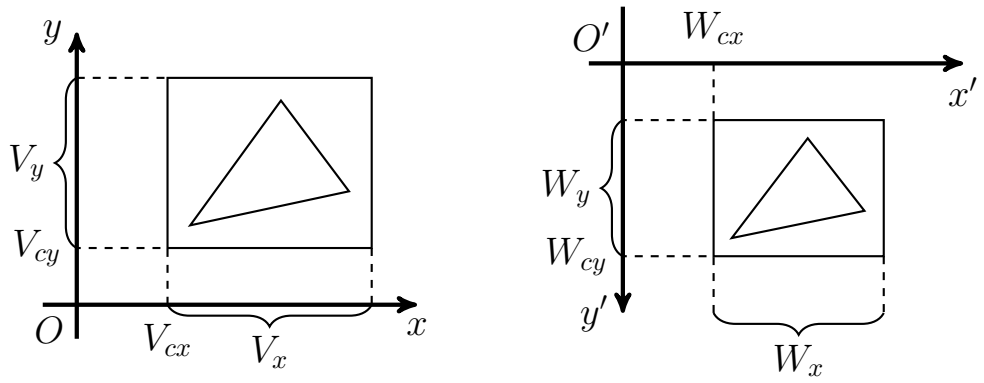
РЕКОМЕНДАЦИИ ДЛЯ ВЫПОЛНЕНИЯ ЗАДАНИЯ 4

1 Предварительные сведения

Пусть в картинной системе координат выделен кадр со следующими параметрами: V_x, V_y — размеры кадра по горизонтали и вертикали соответственно; (V_{cx}, V_{cy}) — координаты нижнего левого угла кадра (см. рис. 1 а)). Стороны кадра параллельны осям координат.



а) картинная плоскость б) экран

Рис. 1: Операция кадрирования

Пусть в системе координат экрана задано прямоугольное окно, с которым необходимо совместить изображение попавшее в кадр. Пусть параметры этого окна следующие: W_x, W_y — размеры окна по горизонтали и вертикали соответственно; (W_{cx}, W_{cy}) — координаты нижнего левого угла окна (см. рис. 1 б)).

Если в картинной системе координат ось Oy направлена вверх, а в СКЭ — вниз, то такое совмещение можно выразить преобразованием:

$$\begin{cases} x' = \dfrac{x - V_{cx}}{V_x} W_x + W_{cx}, \\ y' = W_{cy} - \dfrac{y - V_{cy}}{V_y} W_y. \end{cases} \tag{1}$$

2 Порядок выполнения задания

Функциональность, заложенная при выполнении задания 3 должна сохраниться в настоящем проекте. Поэтому, для начала создадим копию проекта созданного в третьем задании и будем поэтапно вносить изменения в этот проект.

2.1 Изображение прямоугольника

В форме обязан присутствовать прямоугольник, внутри которого будет отрисовываться изображение. Прежде чем изображать его, добавим в форму параметры этого прямоугольника — расстояние от прямоугольника до левой, правой, верхней и нижней границы окна. Для этого в файле `Form1.h` к параметрам класса формы (сейчас добавлен один параметр — `lines`) добавим описание параметров

```
float left, right, top, bottom;
```

Кроме этих параметров для уменьшения количества повторений вычислений введем еще 8 параметров, упомянутых в пункте 1:

```
float Wcx, Wcy, Wx, Wy;
float Vcx, Vcy, Vx, Vy;
```

Здесь первые 4 параметра относятся к прямоугольнику на экране, остальные 4 параметра относятся к прямоугольнику в картинной плоскости: в них будем сохранять параметры, прочитанные из файла.

Добавим начальную инициализацию параметров. Для этого в процедуре загрузки формы (`Form1_Load`) присвоим начальные значения параметрам `left`, `right`, `top` и `bottom`. Паратры `Wcx`, `Wcy`, `Wx` и `Wy` связаны с ними и с формой соотношениями:

```
Wcx = left;
Wcy = Form::ClientRectangle.Height - bottom;
Wx  = Form::ClientRectangle.Width  - left - right;
Wy  = Form::ClientRectangle.Height - top - bottom;
```

Параметры `Wcy`, `Wx` и `Wy` зависят от размеров формы. Поэтому необходимо установить обновление этих значений в обработчике события изменения размера формы (`Form1_Resize`, такой обработчик события добавлялся к форме в первом задании). После их перевычисления нужно дать команду обновления формы.

Теперь в процедуре отрисовки формы (`Form1_Paint`) добавим описание переменной `rectPen` — пера для черчения прямоугольника (желательно, чтобы характеристики этого пера отличались от характеристик пера, которым чертятся отрезки) и начертим сам прямоугольник:

```
g->DrawRectangle(rectPen, Wcx, top, Wx, Wy);
```

Обратите внимание, что третий аргумент `DrawRectangle` — параметр `top`, а не `Wcy`. Этой процедуре в качестве второго и третьего аргументов передаются координаты верхнего левого угла, тогда как параметры `Wcx` и `Wcy` — координаты левого нижнего угла.

Теперь проект можно запустить. На форме должен отрисовываться прямоугольник и его размеры должны меняться при изменении размеров формы.

Желательно изменить свойство *Anchor* кнопки на форме, чтобы при изменении размера формы ее положение было привязано к краю формы. Подберите значения параметров `left`, `right`, `top` и `bottom`, чтобы прямоугольник занимал большую часть формы, но при этом поля должны остаться обозримыми и кнопка не должна наезжать на границы и внутренность прямоугольника.

Кроме того, желательно поменять свойства формы *MinimumSize* для того, чтобы предотвратить уменьшение размера формы и принятия прямоугольником отрицательных размеров.

2.2 Добавление преобразования кадрирования

Параметры `Vcx`, `Vcy`, `Vx`, `Vy`, совместно с формулой (1) задают начальное преобразование для изображения.

Для этого преобразования в файле `Transform.cpp` опишем процедуру

```
void frame (float Vx, float Vy, float Vcx, float Vcy,
            float Wx, float Wy, float Wcx, float Wcy,
            mat c)
```

сохраняющую в параметре `c` матрицу преобразования. Для вычисления матрицы будем использовать элементарные преобразования (для которых уже реализованы процедуры в результате выполнения задания 3).

Преобразование, заданное формулой (1), можно разбить на последовательность «элементарных этапов»:

1. Перенос с коэффициентами $-V_{cx}$ и $-V_{cy}$;
2. Масштабирование с коэффициентами W_x/V_x и W_y/V_y;
3. Зеркальное отражение относительно оси Ox;
4. Перенос с коэффициентами W_{cx} и W_{cy}.

Добавим описание заголовка процедуры `frame` в файл `Transform.h`.

2.3 Чтение входного файла

Формат файла претерпел незначительные изменения: в первой значащей строке должны быть 4 числа — параметры `Vcx`, `Vcy`, `Vx`, `Vy`.

В начале выполнения задания 3, в процедуре открытия файла (`btnOpen_Click`) в качестве матрицы начального преобразования бралась единичная матрица (в результате окончательного выполнения задания 3 это преобразование могло измениться). Теперь констатируем, что начальным преобразованием должно быть преобразование кадрирования с параметрами `Vcx`, `Vcy`, `Vx` и `Vy`, прочитанными из файла, и текущими значениями параметров `Wcx`, `Wcy`, `Wx` и `Wy`.

Итак, внесем в процедуру изменения:

- после открытия файла первые четыре прочитанных числа сохраним в параметрах класса `Vcx`, `Vcy`, `Vx`, `Vy`;
- проинициализируем матрицу `T` с помощью вызова

```
frame (Vx, Vy, Vcx, Vcy, Wx, Wy, Wcx, Wcy, T);
```

К этому же вызову необходимо прибегнуть при нажатии *Esc* процедуре-обработчике нажатия клавиш (восстановление первоначального изображения).

Проект можно снова запустить. Теперь при загрузке файла первоначальное изображение, попадающее на форму, должно зависеть от параметров `Vcx`, `Vcy`, `Vx`, `Vy` в первой значащей строке файла: при загрузке файла изображение сдвигается, переворачивается и масштабируется.

2.4 Масштабирование, связанное с размером формы

По условию, при изменении размеров формы изображение, попавшее в прямоугольник, должно изменять свой масштаб вместе с прямоугольником. Для реализации этого дополним процедуру-обработчик события изменения размера формы (`Form1_Resize`). После выполнения первого пункта задания в этой процедуре меняются значения `Wx` и `Wy`. Перед тем, как они изменяются, сохраним их старые значения:

```
float oldWx = Wx, oldWy = Wy;
```

Зная старые размеры прямоугольника можно вычислить коэффициенты масштабирования изображения.

При растяжении формы координаты левого верхнего угла прямоугольника относительно формы не изменяются. Поэтому целесообразно выполнять масштабирование изображения относительно этой точки.

Таким образом, нужно выполнить преобразование масштабирования относительно точки с координатами (`Wcx`, `top`) с коэффициентами масштабирования `Wx`/`oldWx` и `Wy`/`oldWy`.

2.5 Реализация алгоритма отсечения отрезков

Добавьте в проект файлы `Clip.cpp` и `Clip.h` с описанием процедур, реализующих алгоритм отсечения отрезков, соответствующий вашему варианту.

Основной процедурой в `Clip.cpp` должна быть процедура:

```
bool clip (point &A, point &B, point Pmin, point Pmax)
```

где `A` и `B` — концы отсекаемого отрезка, `Pmin` и `Pmax` — точки соответственно с минимальными и максимальными значениями координат окна отсечения (прямоугольника).

Подключите описанный код к проекту (подключите использование библиотеки "clip.h"с помощью `#include` в основном файле проекта, как это было сделано с `Transform.h` в задании 3).

В файле `Form1.h` внесите изменения в процедуру отрисовки формы (`Form1_Paint`): перед тем как нарисовать отрезок, произведите для него запуск процедуры отсечения и только если останется видимая часть, начертите ее.

Проект можно запустить. Теперь за пределами прямоугольника не должно появляться частей отрезков.

2.6 Добавление наименований отрезков в изображение

Добавьте параметр класса

```
bool drawNames;
```

который будет принимать значение `true`, если на форме должны выводиться имена отрезков, и `false` — в противном случае.

Добавьте инициализации этого параметра в процедуру загрузки формы (`Form1_Load`).

Добавьте обработчик нажатия клавиши P в процедуру обработки нажатия клавиш (`Form1_KeyDown`) для изменения значения параметра `drawNames`.

Внесите изменения в процедуру отрисовки формы (`Form1_Paint`): если `drawNames`, то для каждого видимого отрезка вычислите координаты его середины и рядом с ней выведите его название (пример вывода текста в изображение присутствовал в первом задании).