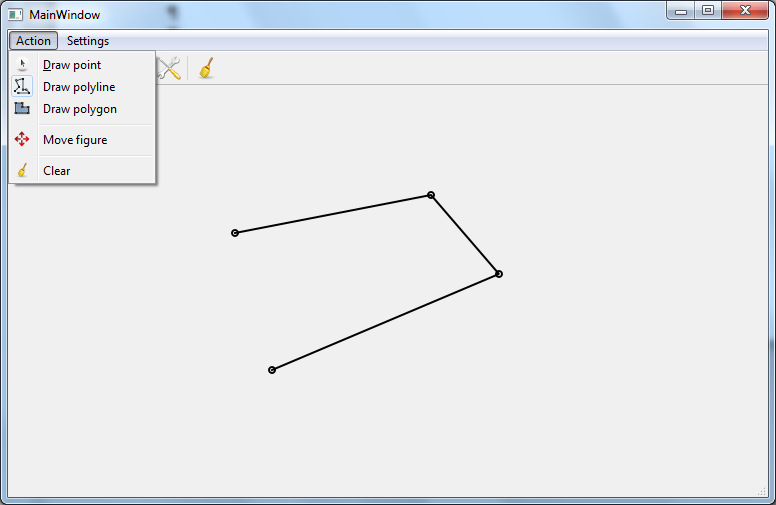
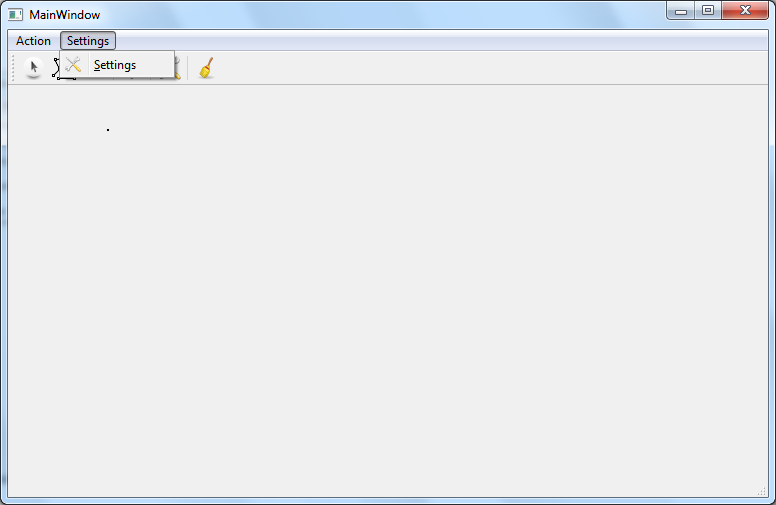
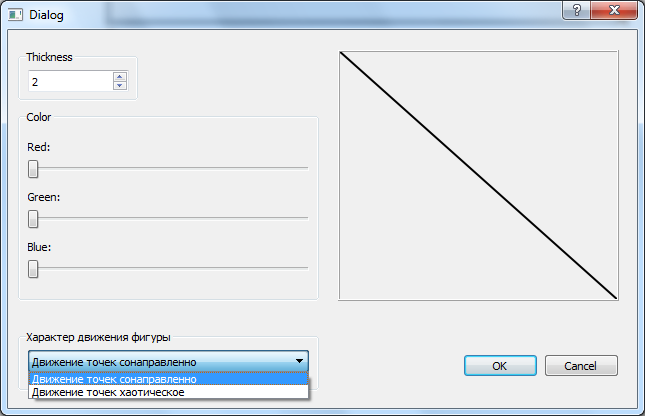
# Задание 1

* Реализовать приложение «Полиморфная кривая», которое обладает следующей функциональностью :

## Внешний вид приложения:







## Описание кнопок

1. Кнопка “Draw points”,
   1. при нажатии которой активизируется режим добавления точек на форму.
   2. Повторное нажатие отключает этот режим.
   3. Одновременно на форме может быть не больше четырех точек. (Если добавляется пятая, то исчезает первая и так далее).
   4. Нарисовать точки можно с помощью функции ***void QPainter::drawPoint(int x, int y);*** Но нам удобнее рисовать точки как эллипсы ***void QPainter::drawEllipse(const QPoint & center, int rx, int ry);*** Радиус можно взять равным 2.
2. Кнопка “Draw polyline”,
   1. при нажатии которой активизируется режим построения полилинии на форму.
   2. Повторное нажатие отключает этот режим.
   3. Функция для построения ***void QPainter::drawPolyline(const QPointF \* points, int pointCount)***, где points – указатель на массив точек, pointCount – количество точек для отрисовки.
3. Кнопка “Draw polygon”,
   1. при нажатии которой активизируется режим построения полигона на форму.
   2. Повторное нажатие отключает этот режим.
   3. Функция для построения ***void QPainter::drawPolygon(const QPointF \* points, int pointCount, Qt::FillRule fillRule = Qt::OddEvenFill)***, где points – указатель на массив точек, pointCount – количество точек для отрисовки.
4. Кнопка “Move figure”,
   1. при нажатии которой активизируется режим движения фигуры по виджету.
   2. Повторное нажатие отключает этот режим.
   3. Точки могут двигаться как сонаправленно, так и хаотически.
   4. Не допускается выход точки за пределы виджета.
5. Кнопка “Settings”,
   1. при нажатии которой открывается диалоговое окно, где пользователем устанавливается толщина и цвет пера, рисующего точки и фигуры.
6. Кнопка “Clear”,
   1. при нажатии которой все точки удаляются с виджета и из контейнера.

## «Архитектура» приложения.

* Точки храним в массиве QPoint.
* Все функции должны быть реализованы с помощью объектов QAction.
* Доступ к функциям должен быть через меню и toolbar ( QPushButton не используем)
* Можно проверять, нажата ли кнопка, с помощью ***bool QAction::isChecked() const***
* Устанавливать или снимать флаг (визуально это отображается «утоплена» кнопка или нет) с помощью ***void QAction::setChecked(bool).*** 
* Точки добавляются в массив в очередную ячейку при клике на виджет и активной кнопке “Draw points”; Для реакции приложения на клики мышкой необходимо перегрузить метод mousePressEvent. После добавления точки в массив необходимо перерисовать форму.
* В statusBar необходимо отобразить какой режим выбрал пользователь.
* В диалоговом окне необходимо отображать текущие параметры пера и режим рисования.
* Работа с QComboBox (виджет, отображающий выбор режима движения точек)
  + ***void QComboBox::setCurrentIndex(int index) –*** установка строки по индексу;
  + ***int QComboBox currentIndex() const –*** получение текущего индекса.

## Параметры рисования

* Рисование фигуры осуществляется в paintEvent. Для того, чтобы программа понимала, что ей рисовать (полигон или полилинию), удобно ввести необходимое количество флагов.
* Полилинии и полигоны рисуются всегда, но в зависимости от **количества созданных,** а не от размера массива (т.е. можно нарисовать линию и из 3-х точек).

## Движение фигуры

* Движение фигур реализуется как
  + изменение координат точек
  + и перерисовка формы.
* Достаточно будет выбрать случайным образом одно из четырех направлений (вправо, влево, вверх, вниз). Координаты всех точек изменяются на одно и то же значение (шаг).
* Для того чтобы фигура двигалась, необходимо перегрузить метод timerEvent. За каждый промежуток времени точки приобретают новые координаты.
* Обработчик кнопки “Move figure”:

void MainWindow::move\_figure()

{

if (таймер не запущен)

{

timestep = 50; //интервал таймера

//запускаем таймер

//устанавливаем признак того, что таймер запущен

}

else

{

//устанавливаем признак того, что таймер остановлен;

//Останавливаем таймер;

}

}

* Режим движения точек ( сонаправленный или хаотический ) можно возвращать из диалога булевской переменной)