Name:	Mustafa Chowdhury
Student access ID:	Ge3306
Date (MM/DD/YYYY):	04/21/2019
Group Number (if any):	2

Title of the change request	
Sources:	**Source#1:** https://doc.qt.io/qt-5/qpainter.html **Source#2:**

1. Change Request and concepts*:* (10 points)

Change Request: "Add a new convex pentagon selection instrument feature for selecting images. This new instrument shall meet all the features of the original selection instrument. You must use proper icons and shortcut for this selection instrument."

From the change request, I extract following concept:

1) Add – is used to communication, therefore it is an irrelevant concept.
2) Convex Pentagon – need to implement this selection instrument in the program, therefore it is an external concept
3) Selection: Some form of code already implemented in the program, therefore it a relevant concept.
4) Instrument: Some form of code already implemented in the program, therefore it a relevant concept.

Table 1 Significant Concepts and their details

SN#	Concept Name	Details of how your extracted this concept.
CON1	Selection	Some from of code already implemented in the program, therefore it a relevant concept.
CON2	Instrument	Some from of code already implemented in the program, therefore it a relevant concept.

2. Functional requirements: (10 points)

Table 2 Functional Requirements

Requirement#	Functional Requirement Details
FR1	Image shall need to be cropped as a convex pentagon shape from the canvas.
FR2	Cropped convex pentagon shaped image shall be paste in any window of the easy paint program.
FR3	User can select a convex pentagon selection icon from easyPaint interface and using shortcut.

3. Concept Location:

Methodology: (5 points)

After extracting the significant concepts from change request, first I select the relevant concept "Selection" to search. I used grep search and choose to search query on entire easy paint project. I found several matches. Based on the matches, I analyzed the code to find the concept location. Because this change request demands a new functionality, a new class need to be created and it shall derive from the AbstractSelection class, which is the concept location.

3.1 Dependency Search (Use this section if you have used dependency search) (7 points)

N/A

Table 3 if Dependency Search is used

Class/file name	Tool used	Mark	Explanation

Partial Class Dependency Graph (UML): (3 points)

N/A

3.2 Grep Search (Use this section if you have used grep search) (10 points)

Table 4 If Grep Search is used

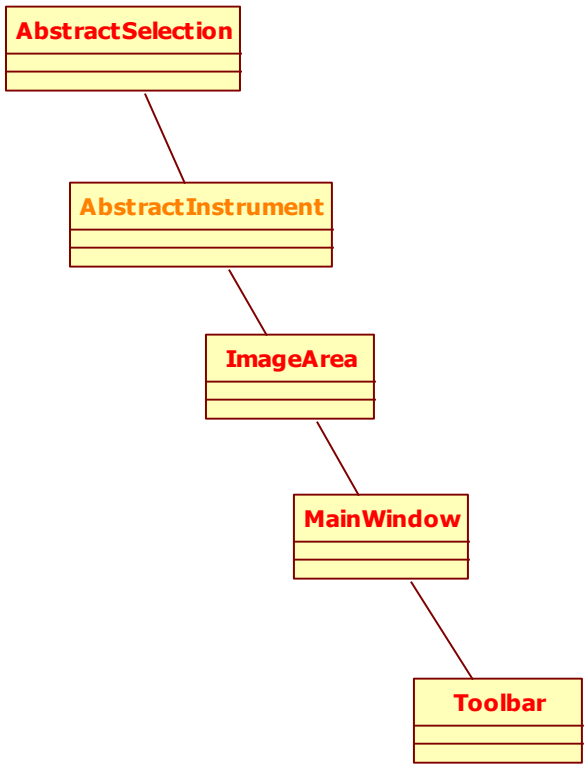
Concept	Query	#Results	Target class/file	Tool used	Explanation
Selection Instrument	*Selection*	Found	AbstractSelection	Grep Search	To constitute the convex pentagon selection instrument functionality, a derived class shall be created from AbstractSelection class, therefore concept location is found.

4. Impact Analysis: (10 points)

Table 5 The list of all the classes visited during impact analysis

Class name	Tool used	Mark	Explanation
AbstractSelection	View all References	Impacted	Initial impact set.
AbstractInstrument	View all References	Propagating	Propagate information from AbstractSlection class to ImageArea
ImageArea	View all References	Impacted	Instrument handler for new selection instrument need to be added
easyPaintEnum.h	View all References	impacted	Enum type for new selection instrument need to add.
MainWindow	View all References	impacted	New selection instrument icon and action bar need to be implemented.
Toolbar	View all References	impacted	New selection instrument button need to be added into the instrument toolbar.

Partial class interaction graph (use starUML): (5 points)



5. Prefactoring: (5 points)

Table 6 Prefactoring Code Files

File Name	Refactoring Issue	Lines of Code		
		Added	Deleted	Total

N/A

6. Actualization: (10 points)

Table 7 Actualization Summary

Code Files					
Visited#	**Changed#**	**Added#**	**Propagating#**	**Unchanged#**	**Added to Changed Set#**
7	6	6	1	0	6

Table 8 Actualization Code Files

File Name	Task	Lines of Code		
		Added	**Deleted**	**Total**
ConvexPentagonSelectionInstrument.h	Added a new class derived from AbstractSelection class with new functionalities	72	0	72
ConvexPentagonSelectionInstrument.cpp	Implement functionality of header file	208	0	208
AbstractSelection.cpp	Inside of DrawBordar() function convex pentagon drawing implemented	20	0	20
easyPaintEnum.h	Enum type of convex pentagon selection instrument is added	1	0	1
ImageArea.cpp	Instrument handler for convex pentagon is added	30	0	30
MainWindow.cpp	Convex Pentagon selection instrument icon	9	0	9

	and action is added			
Resources.qrc	Icons path is added	1	0	1
ToolBar.h	Variable for the pentagon button added	6	0	6
ToolBar.cpp	Layout for the Convex Pentagon Selection Instrument icon implemented	6	0	6

7. Postfactoring: (5 points)

Table 9 Postfactoring Code Files

File Name	Refactoring Issue	Lines of Code		
		Added	Deleted	Total

N/A

8. Verification: (15 points)

Only functional testing is performed to verify the functionality of the program match with the change request.

Table 10 Statement Verification Summary

File Name	Coverage of Application			Unit Test Failed (Just indicate the SN#)	Bugs Found
	Total statements added	Total statements covered	Statement coverage %		
Example: TestMe.cpp	7	5	71%	UT#1	0

Unit Test Case Details:

N/A

Functional Test Case Details:

After implementing the change request, I draw an eclipse inside of a rectangle on the canvas area and filled with two different colors. I selected convex pentagon selection icon and select a convex pentagon shape from the image I drawn. The convex pentagon shape cut properly from the drawn image. After that, I opened a new window and paste the image that I cut. This work properly only one bug is found, which is the fifthPoint of the pentagon shape is not filled. I spent lot of time and couldn't figure it out. I add this bug into my product backlog and fix this bug in the future. I also check for the old functionality after modification. And all the other functions working properly and does not impacted by the change.

9. Highlighted Source Code: (10 points)

ConvexPentagonSelectionInstrument:

```
// Mustafa
#ifndef ConvexPentagonSelectionInstrument_H
#define ConvexPentagonSelectionInstrument_H

#include "abstractselection.h"

QT_BEGIN_NAMESPACE
class QUndoStack;
QT_END_NAMESPACE

class ConvexPentagonSelectionInstrument : public AbstractSelection
{
        Q_OBJECT

public:
        explicit ConvexPentagonSelectionInstrument(QObject *parent = 0);

        /**
        * @brief Clears background image at selection area.
        *
        * @param imageArea ImageArea for applying changes.
        */
        void clearSelectionBackground(ImageArea &imageArea);
        /**
        * @brief Copying image to the clipboard.
        *
        * @param imageArea ImageArea for applying changes.
        */
        void copyImage(ImageArea &imageArea);
        /**
        * @brief Paste image from the clipboard.
        *
        * @param imageArea ImageArea for applying changes.
        */
        void pasteImage(ImageArea &imageArea);
        /**
        * @brief Cut image to the clipboard.
        *
        * @param imageArea ImageArea for applying changes.
        */
        void cutImage(ImageArea &imageArea);


private:
        void startAdjusting(ImageArea &imageArea);
        void startSelection(ImageArea &);
        void startResizing(ImageArea &imageArea);
        void startMoving(ImageArea &imageArea);
        void select(ImageArea &);
        void resize(ImageArea &);
        void move(ImageArea &);
        void completeSelection(ImageArea &imageArea);
```

```
        void completeResizing(ImageArea &imageArea);
        void completeMoving(ImageArea &);
        void clear();
        void paint(ImageArea &imageArea, bool = false, bool = false);
        void showMenu(ImageArea &);


        QImage mSelectedImage, /**< Copy of selected image. */
               mPasteImage; /**< Image to paste */
        QPolygon mScaleneSelect;

signals:
        void sendEnableCopyCutActions(bool enable);
        void sendEnableSelectionInstrument(bool enable);

};

#endif // ConvexPentagonSelectionInstrument_H

// MUSTAFA

#include "ConvexPentagonSelectionInstrument.h"
#include "../imagearea.h"
#include "../undocommand.h"
#include "math.h"

#include <QPainter>
#include <QApplication>
#include <QClipboard>

ConvexPentagonSelectionInstrument::ConvexPentagonSelectionInstrument(QObject *parent) :
        AbstractSelection(parent)
{
}

void ConvexPentagonSelectionInstrument::copyImage(ImageArea &imageArea)
{
        if (mIsSelectionExists)
        {
                imageArea.setImage(mImageCopy);
                QClipboard *globalClipboard = QApplication::clipboard();
                QImage copyImage;
                if (mIsImageSelected)
                {
                        copyImage = mSelectedImage;
                }
                else
                {
                        copyImage = imageArea.getImage()->copy(mTopLeftPoint.x(),
mTopLeftPoint.y(), mWidth, mHeight);
                }
                globalClipboard->setImage(copyImage, QClipboard::Clipboard);
        }
}

void ConvexPentagonSelectionInstrument::cutImage(ImageArea &imageArea)
{
```

```
        if (mIsSelectionExists)
        {
                copyImage(imageArea);
                if (mIsSelectionExists)
                {
                        imageArea.setImage(mImageCopy);
                        paint(imageArea);
                }
                makeUndoCommand(imageArea);
                if (/*mSelectedImage != mPasteImage || !*/mIsImageSelected)
                {
                        imageArea.setImage(mImageCopy);
                }
                else
                {
                        clearSelectionBackground(imageArea);
                }
                mTopLeftPoint = QPoint(0, 0);
                mBottomRightPoint = QPoint(0, 0);
                mImageCopy = *imageArea.getImage();
                imageArea.update();
                mIsSelectionExists = false;
                imageArea.restoreCursor();
                emit sendEnableCopyCutActions(false);
        }
}

void ConvexPentagonSelectionInstrument::pasteImage(ImageArea &imageArea)
{
        QClipboard *globalClipboard = QApplication::clipboard();
        if (mIsSelectionExists)
        {
                imageArea.setImage(mImageCopy);
                paint(imageArea);
                mImageCopy = *imageArea.getImage();
        }
        makeUndoCommand(imageArea);
        mPasteImage = globalClipboard->image();
        if (!mPasteImage.isNull())
        {
                mSelectedImage = mPasteImage;
                mImageCopy = *imageArea.getImage();
                mTopLeftPoint = QPoint(0, 0);
                mBottomRightPoint = QPoint(mPasteImage.width(), mPasteImage.height()) -
QPoint(1, 1);
                mHeight = mPasteImage.height();
                mWidth = mPasteImage.width();
                mIsImageSelected = mIsSelectionExists = true;
                paint(imageArea);
                drawBorder(imageArea);
                imageArea.restoreCursor();
                emit sendEnableCopyCutActions(true);
        }
}


void ConvexPentagonSelectionInstrument::startAdjusting(ImageArea & imageArea)
{
```

```
        mImageCopy = *imageArea.getImage();
        mIsImageSelected = false;
}

void ConvexPentagonSelectionInstrument::startSelection(ImageArea &)
{
}

void ConvexPentagonSelectionInstrument::startResizing(ImageArea &imageArea)
{
        if (!mIsImageSelected)
        {
                clearSelectionBackground(imageArea);
        }
}

void ConvexPentagonSelectionInstrument::startMoving(ImageArea &imageArea)
{
        clearSelectionBackground(imageArea);
}

void ConvexPentagonSelectionInstrument::select(ImageArea &)
{
}

void ConvexPentagonSelectionInstrument::resize(ImageArea &)
{
}

void ConvexPentagonSelectionInstrument::move(ImageArea &)
{
}

void ConvexPentagonSelectionInstrument::completeSelection(ImageArea &imageArea)
{
        mSelectedImage = imageArea.getImage()->copy(mTopLeftPoint.x(),
                mTopLeftPoint.y(),
                mWidth, mHeight);
        emit sendEnableCopyCutActions(true);
}

void ConvexPentagonSelectionInstrument::completeResizing(ImageArea &imageArea)
{
        mSelectedImage = imageArea.getImage()->copy(mTopLeftPoint.x(),
                mTopLeftPoint.y(),
                mWidth, mHeight);
}
void ConvexPentagonSelectionInstrument::completeMoving(ImageArea &)
{

}
// add
void ConvexPentagonSelectionInstrument::clearSelectionBackground(ImageArea &imageArea)
{

        QPainter blankPainter(imageArea.getImage());
```

```
        blankPainter.setPen(Qt::white);
        blankPainter.setBrush(QBrush(Qt::white));
        blankPainter.setBackgroundMode(Qt::OpaqueMode);

        //copied justine code from CR#2 to draw convex oentagon
        QPolygon *poly = new QPolygon(QRect(mTopLeftPoint, mBottomRightPoint), true);
        QPoint *fifthPoint = new QPoint(((int)mTopLeftPoint.rx() +
(int)mBottomRightPoint.rx()) / 2,
                ((int)mTopLeftPoint.ry() / 4));
        poly->setPoint(0, *fifthPoint);

        blankPainter.drawPolygon(*poly);
        blankPainter.end();
        mImageCopy = *imageArea.getImage();

}

void ConvexPentagonSelectionInstrument::clear()
{
        mSelectedImage = QImage();
        emit sendEnableCopyCutActions(false);
}
//add
void ConvexPentagonSelectionInstrument::paint(ImageArea &imageArea, bool, bool)
{
        if (mIsSelectionExists)
        {
                if (mTopLeftPoint != mBottomRightPoint)
                {
                        //mustafa
                        //copied justine code from CR#2 to draw convex oentagon
                        QPainter painter(imageArea.getImage());
                        //QPolygon *poly = new QPolygon(QRect(mTopLeftPoint,
mBottomRightPoint), true);
                        //QPoint *fifthPoint = new QPoint(((int)mTopLeftPoint.rx() +
(int)mBottomRightPoint.rx()) / 2,
                        //      ((int)mTopLeftPoint.ry() / 4));
                        //poly->setPoint(0, *fifthPoint);

                        //crop the image into a  region of it
                        //painter.setClipRegion(QRegion(*poly));
                        QRect source(0, 0, mSelectedImage.width(), mSelectedImage.height());
                        QRect target(mTopLeftPoint, mBottomRightPoint);
                        painter.drawImage(target, mSelectedImage, source);
                        painter.end();
                }
                imageArea.setEdited(true);
                imageArea.update();
        }
}

void ConvexPentagonSelectionInstrument::showMenu(ImageArea &)
{
}
```

Rest of the modification is showed by git diff:

```
musta@DESKTOP-ST5QVK3 MINGW64 ~/Desktop/Good/csc4111w19grp2/Project
(master)
$ git diff
diff --git a/Project/CMakeLists.txt b/Project/CMakeLists.txt
index c611259..591823f 100644
--- a/Project/CMakeLists.txt
+++ b/Project/CMakeLists.txt
@@ -61,7 +61,8 @@ set (HEADERS
     sources/instruments/irregularpentagoninstrument.h
     sources/instruments/regularpentagoninstrument.h
     sources/instruments/scaleneobtusetriangleinstrument.h
+       sources/instruments/convexpentagonselectioninstrument.h)


 #------- sources --------
@@ -112,7 +113,8 @@ set (SOURCES
     sources/instruments/irregularpentagoninstrument.cpp
     sources/instruments/regularpentagoninstrument.cpp
     sources/instruments/scaleneobtusetriangleinstrument.cpp
+       sources/instruments/convexpentagonselectioninstrument.cpp)

 #------- resources -------
 set (RESOURCE_PATH
diff --git a/Project/sources/easypaintenums.h
b/Project/sources/easypaintenums.h
index 3f7099d..7fe736d 100644
--- a/Project/sources/easypaintenums.h
+++ b/Project/sources/easypaintenums.h
@@ -59,6 +59,9 @@ typedef enum
     TEXT,
       SCALENECURSOR,

+       //mustafa
+       CONVEXPENTAGONCURSOR,
+
     // Don't use it. (Used to know count of current instrument)
     INSTRUMENTS_COUNT
 } InstrumentsEnum;
diff --git a/Project/sources/imagearea.cpp
b/Project/sources/imagearea.cpp
index be2e404..62475b5 100644
--- a/Project/sources/imagearea.cpp
+++ b/Project/sources/imagearea.cpp
@@ -48,6 +48,7 @@
 #include "instruments/irregularpentagoninstrument.h"
 #include "instruments/scaleneobtusetriangleinstrument.h"
 #include "instruments/scaleneselectioninstrument.h"
+#include "instruments/convexpentagonselectioninstrument.h"

 #include "dialogs/resizedialog.h"

@@ -149,10 +150,18 @@ ImageArea::ImageArea(const bool &isOpen, const
QString &filePath, QWidget *paren
```

```
        connect(scaleneSelectionInstrument,
SIGNAL(sendEnableCopyCutActions(bool)), this,
SIGNAL(sendEnableCopyCutActions(bool)));
        connect(scaleneSelectionInstrument,
SIGNAL(sendEnableSelectionInstrument(bool)), this,
SIGNAL(sendEnableSelectionInstrument(bool)));

+       //Mustafa
+       ConvexPentagonSelectionInstrument
*convexPentagonSelectionInstrument = new
ConvexPentagonSelectionInstrument(this);
+       connect(convexPentagonSelectionInstrument,
SIGNAL(sendEnableCopyCutActions(bool)), this,
SIGNAL(sendEnableCopyCutActions(bool)));
+       connect(convexPentagonSelectionInstrument,
SIGNAL(sendEnableSelectionInstrument(bool)), this,
SIGNAL(sendEnableSelectionInstrument(bool)));
+
     // Instruments handlers
     mInstrumentsHandlers.fill(0, (int)INSTRUMENTS_COUNT);
     mInstrumentsHandlers[CURSOR] = selectionInstrument;
        mInstrumentsHandlers[SCALENECURSOR] =
scaleneSelectionInstrument; //JUSTIN
+       // mustafa
+       mInstrumentsHandlers[CONVEXPENTAGONCURSOR] =
convexPentagonSelectionInstrument;
+
     mInstrumentsHandlers[PEN] = new PencilInstrument(this);
     mInstrumentsHandlers[LINE] = new LineInstrument(this);
     mInstrumentsHandlers[ERASER] = new EraserInstrument(this);
@@ -384,6 +393,12 @@ void ImageArea::copyImage()
                ScaleneSelectionInstrument *instrument = static_cast
<ScaleneSelectionInstrument*>  (mInstrumentsHandlers.at(SCALENECURSOR));
                instrument->copyImage(*this);
        }
+       //mustafa
+       else if (DataSingleton::Instance()->getInstrument() ==
CONVEXPENTAGONCURSOR)
+       {
+               ConvexPentagonSelectionInstrument *instrument =
static_cast <ConvexPentagonSelectionInstrument*>
(mInstrumentsHandlers.at(CONVEXPENTAGONCURSOR));
+               instrument->copyImage(*this);
+       }

 }

@@ -396,6 +411,13 @@ void ImageArea::pasteImage()
                instrument->pasteImage(*this);
                return;
        }
+       //mustafa
+       if (DataSingleton::Instance()->getInstrument() ==
CONVEXPENTAGONCURSOR)
+       {
+               ConvexPentagonSelectionInstrument *instrument =
static_cast <ConvexPentagonSelectionInstrument*>
(mInstrumentsHandlers.at(CONVEXPENTAGONCURSOR));
```

```
+                instrument->pasteImage(*this);
+                return;
+        }
         if (DataSingleton::Instance()->getInstrument() != CURSOR)
                 emit sendSetInstrument(CURSOR);
         SelectionInstrument *instrument = static_cast
<SelectionInstrument*> (mInstrumentsHandlers.at(CURSOR));
@@ -415,6 +437,12 @@ void ImageArea::cutImage()
                 ScaleneSelectionInstrument *instrument = static_cast
<ScaleneSelectionInstrument*>  (mInstrumentsHandlers.at(SCALENECURSOR));
                 instrument->cutImage(*this);
         }
+        //mustafa
+        else if (DataSingleton::Instance()->getInstrument() ==
CONVEXPENTAGONCURSOR)
+        {
+                ConvexPentagonSelectionInstrument *instrument =
static_cast <ConvexPentagonSelectionInstrument*>
(mInstrumentsHandlers.at(CONVEXPENTAGONCURSOR));
+                instrument->cutImage(*this);
+        }
 }

 void ImageArea::mousePressEvent(QMouseEvent *event)
@@ -520,6 +548,8 @@ void ImageArea::restoreCursor()
          break;
      case CURSOR:
         case SCALENECURSOR: //JUSTIN
+        //mustafa
+        case CONVEXPENTAGONCURSOR:
          mCurrentCursor = new QCursor(Qt::CrossCursor);
          setCursor(*mCurrentCursor);
          break;
@@ -570,7 +600,7 @@ void ImageArea::drawCursor()
         case FILL: case RECTANGLE: case CONCAVEPENTAGON: case
CONVEXPENTAGON: case SCALENEOBTUSETRIANGLE: case ELLIPSE:
+        sources/instruments/scaleneselectioninstrument.h
+        sources/instruments/convexpentagonselectioninstrument.h)


 #------- sources --------
@@ -112,7 +113,8 @@ set (SOURCES
     sources/instruments/irregularpentagoninstrument.cpp
     sources/instruments/regularpentagoninstrument.cpp
     sources/instruments/scaleneobtusetriangleinstrument.cpp
+        sources/instruments/convexpentagonselectioninstrument.cpp)

 #------- resources -------
 set (RESOURCE_PATH
diff --git a/Project/sources/easypaintenums.h
b/Project/sources/easypaintenums.h
index 3f7099d..7fe736d 100644
--- a/Project/sources/easypaintenums.h
+++ b/Project/sources/easypaintenums.h
@@ -59,6 +59,9 @@ typedef enum
     TEXT,
         SCALENECURSOR,
```

```
+       //mustafa
+       CONVEXPENTAGONCURSOR,
+
     // Don't use it. (Used to know count of current instrument)
     INSTRUMENTS_COUNT
 } InstrumentsEnum;
diff --git a/Project/sources/imagearea.cpp
b/Project/sources/imagearea.cpp
index be2e404..62475b5 100644
--- a/Project/sources/imagearea.cpp
+++ b/Project/sources/imagearea.cpp
@@ -48,6 +48,7 @@
 #include "instruments/irregularpentagoninstrument.h"
 #include "instruments/scaleneobtusetriangleinstrument.h"
 #include "instruments/scaleneselectioninstrument.h"
+#include "instruments/convexpentagonselectioninstrument.h"

 #include "dialogs/resizedialog.h"

@@ -149,10 +150,18 @@ ImageArea::ImageArea(const bool &isOpen, const
QString &filePath, QWidget *paren
         connect(scaleneSelectionInstrument,
SIGNAL(sendEnableCopyCutActions(bool)), this,
SIGNAL(sendEnableCopyCutActions(bool)));
         connect(scaleneSelectionInstrument,
SIGNAL(sendEnableSelectionInstrument(bool)), this,
SIGNAL(sendEnableSelectionInstrument(bool)));

+       //Mustafa
+       ConvexPentagonSelectionInstrument
*convexPentagonSelectionInstrument = new
ConvexPentagonSelectionInstrument(this);
+       connect(convexPentagonSelectionInstrument,
SIGNAL(sendEnableCopyCutActions(bool)), this,
SIGNAL(sendEnableCopyCutActions(bool)));
+       connect(convexPentagonSelectionInstrument,
SIGNAL(sendEnableSelectionInstrument(bool)), this,
SIGNAL(sendEnableSelectionInstrument(bool)));
+
     // Instruments handlers
     mInstrumentsHandlers.fill(0, (int)INSTRUMENTS_COUNT);
     mInstrumentsHandlers[CURSOR] = selectionInstrument;
        mInstrumentsHandlers[SCALENECURSOR] =
scaleneSelectionInstrument; //JUSTIN
+       // mustafa
+       mInstrumentsHandlers[CONVEXPENTAGONCURSOR] =
convexPentagonSelectionInstrument;
+
     mInstrumentsHandlers[PEN] = new PencilInstrument(this);
     mInstrumentsHandlers[LINE] = new LineInstrument(this);
     mInstrumentsHandlers[ERASER] = new EraserInstrument(this);
@@ -384,6 +393,12 @@ void ImageArea::copyImage()
                 ScaleneSelectionInstrument *instrument = static_cast
<ScaleneSelectionInstrument*>  (mInstrumentsHandlers.at(SCALENECURSOR));
                 instrument->copyImage(*this);
         }
+       //mustafa
```

```
+       else if (DataSingleton::Instance()->getInstrument() ==
CONVEXPENTAGONCURSOR)
+       {
+               ConvexPentagonSelectionInstrument *instrument =
static_cast <ConvexPentagonSelectionInstrument*>
(mInstrumentsHandlers.at(CONVEXPENTAGONCURSOR));
+               instrument->copyImage(*this);
+       }

 }

@@ -396,6 +411,13 @@ void ImageArea::pasteImage()
                instrument->pasteImage(*this);
                return;
        }
+       //mustafa
+       if (DataSingleton::Instance()->getInstrument() ==
CONVEXPENTAGONCURSOR)
+       {
+               ConvexPentagonSelectionInstrument *instrument =
static_cast <ConvexPentagonSelectionInstrument*>
(mInstrumentsHandlers.at(CONVEXPENTAGONCURSOR));
+               instrument->pasteImage(*this);
+               return;
+       }
        if (DataSingleton::Instance()->getInstrument() != CURSOR)
                emit sendSetInstrument(CURSOR);
        SelectionInstrument *instrument = static_cast
<SelectionInstrument*> (mInstrumentsHandlers.at(CURSOR));
@@ -415,6 +437,12 @@ void ImageArea::cutImage()
                ScaleneSelectionInstrument *instrument = static_cast
<ScaleneSelectionInstrument*>  (mInstrumentsHandlers.at(SCALENECURSOR));
                instrument->cutImage(*this);
        }
+       //mustafa
+       else if (DataSingleton::Instance()->getInstrument() ==
CONVEXPENTAGONCURSOR)
+       {
+               ConvexPentagonSelectionInstrument *instrument =
static_cast <ConvexPentagonSelectionInstrument*>
(mInstrumentsHandlers.at(CONVEXPENTAGONCURSOR));
+               instrument->cutImage(*this);
+       }
 }

 void ImageArea::mousePressEvent(QMouseEvent *event)
@@ -520,6 +548,8 @@ void ImageArea::restoreCursor()
         break;
     case CURSOR:
        case SCALENECURSOR: //JUSTIN
+       //mustafa
+       case CONVEXPENTAGONCURSOR:
         mCurrentCursor = new QCursor(Qt::CrossCursor);
         setCursor(*mCurrentCursor);
         break;
@@ -588,6 +618,8 @@ void ImageArea::drawCursor()
        case CONVEXHEXAGON: case CONCAVEHEXAGON: case REGULARPENTAGON:
case IRREGULARPENTAGON:
```

```
                //jacob
**********************************************************************
**********************************************************************
        case SCALENEOBTUSETRIANGLE: case SCALENECURSOR:
+       // mustafa
+       case CONVEXPENTAGONCURSOR:
         break;
     case PEN:
         if(mRightButtonPressed)
diff --git a/Project/sources/instruments/abstractselection.cpp
b/Project/sources/instruments/abstractselection.cpp
index 99e13fa..05244d6 100644
--- a/Project/sources/instruments/abstractselection.cpp
+++ b/Project/sources/instruments/abstractselection.cpp
@@ -221,6 +221,7 @@ void AbstractSelection::drawBorder(ImageArea
&imageArea)
                        painter.end();
                        imageArea.update();
                }
+
                else if (DataSingleton::Instance()->getInstrument() ==
SCALENECURSOR)
                {
                        QPainter painter(imageArea.getImage());
@@ -242,7 +243,28 @@ void AbstractSelection::drawBorder(ImageArea
&imageArea)
                        painter.end();
                        imageArea.update();
                }
+               // Mustafa
+               else if (DataSingleton::Instance()->getInstrument() ==
CONVEXPENTAGONCURSOR)
+               {
+                       QPainter painter(imageArea.getImage());
+                       painter.setPen(QPen(Qt::blue, 1, Qt::DashLine,
Qt::RoundCap, Qt::RoundJoin));
+                        painter.setBackgroundMode(Qt::TransparentMode);
+                       if (mTopLeftPoint != mBottomRightPoint)
+                       {
+                               // copied Justin code from CR#2, to
draw convex pentagon
+                               QPolygon *poly = new
QPolygon(QRect(mTopLeftPoint, mBottomRightPoint), true);
+
+                               QPoint *fifthPoint = new
QPoint(((int)mTopLeftPoint.rx() + (int)mBottomRightPoint.rx()) / 2,
+                                       ((int)mTopLeftPoint.ry() / 4));

+                               poly->setPoint(0, *fifthPoint);
+
+                               painter.drawConvexPolygon(*poly);
+                       }
+                       imageArea.setEdited(true);
+                       painter.end();
+                       imageArea.update();
+               }
        }
 }
```

```
diff --git a/Project/sources/mainwindow.cpp
b/Project/sources/mainwindow.cpp
index 23d32b5..5775cf8 100644
--- a/Project/sources/mainwindow.cpp
+++ b/Project/sources/mainwindow.cpp
@@ -393,6 +393,13 @@ void MainWindow::initializeMainMenu()
         mSelInstruMenu->addAction(mScaleneCursorAction);
         mInstrumentsActMap.insert(SCALENECURSOR,  mScaleneCursorAction);

+        // mustafa
+        QAction *mConvexPentagonCursorAction = new QAction(tr("Convex
Pentagon Selection"), this);
+        mConvexPentagonCursorAction->setCheckable(true);
+        mConvexPentagonCursorAction-
>setIcon(QIcon(":/media/instruments-
icons/convexPentagonSelection.png"));
+        connect(mConvexPentagonCursorAction, SIGNAL(triggered(bool)),
this, SLOT(instumentsAct(bool)));
+        mSelInstruMenu->addAction(mConvexPentagonCursorAction);
+        mInstrumentsActMap.insert(CONVEXPENTAGONCURSOR,
mConvexPentagonCursorAction);



@@ -759,6 +766,8 @@ void MainWindow::updateShortcuts()

     mInstrumentsActMap[CURSOR]-
>setShortcut(DataSingleton::Instance()-
>getInstrumentShortcutByKey("Cursor"));
         mInstrumentsActMap[SCALENECURSOR]-
>setShortcut(DataSingleton::Instance()-
>getInstrumentShortcutByKey("ScaleneCursor"));//JUSTIN
+        //mustafa
+        mInstrumentsActMap[CONVEXPENTAGONCURSOR]-
>setShortcut(DataSingleton::Instance()-
>getInstrumentShortcutByKey("ConvexPentagonCursor"));
     mInstrumentsActMap[ERASER]-
>setShortcut(DataSingleton::Instance()-
>getInstrumentShortcutByKey("Lastic"));
     mInstrumentsActMap[COLORPICKER]-
>setShortcut(DataSingleton::Instance()-
>getInstrumentShortcutByKey("Pipette"));
     mInstrumentsActMap[MAGNIFIER]-
>setShortcut(DataSingleton::Instance()-
>getInstrumentShortcutByKey("Loupe"));
diff --git a/Project/sources/resources.qrc
b/Project/sources/resources.qrc
index 8e05128..31063bc 100644
--- a/Project/sources/resources.qrc
+++ b/Project/sources/resources.qrc
@@ -45,5 +45,7 @@
          <file>media/instruments-icons/curve.png</file>
          <file>media/actions-icons/clear-gray.png</file>
          <file>media/instruments-icons/scalenecursor.png</file>
+         <file>media/instruments-
icons/convexPentagonSelection.png</file>
+
```

```
     </qresource>
 </RCC>
diff --git a/Project/sources/widgets/toolbar.cpp
b/Project/sources/widgets/toolbar.cpp
index 3613b48..0e99862 100644
--- a/Project/sources/widgets/toolbar.cpp
+++ b/Project/sources/widgets/toolbar.cpp
@@ -77,6 +77,9 @@ void ToolBar::initializeItems()

         //JUSTIN: added button for scalene selection
         mScaleneCursorButton =
createToolButton(mActMap[SCALENECURSOR]);
+
+       //mustafa
+       mConvexPentagonCursorButton =
createToolButton(mActMap[CONVEXPENTAGONCURSOR]);

     mEllipseButton = createToolButton(mActMap[ELLIPSE]);
     mCurveButton = createToolButton(mActMap[CURVELINE]);
@@ -111,6 +114,8 @@ void ToolBar::initializeItems()
         bLayout->addWidget(mConvexHexagonButton, 9, 0);
         //Justin
         bLayout->addWidget(mScaleneCursorButton, 9, 1);
+       //mustafa
+       bLayout->addWidget(mConvexPentagonCursorButton, 10, 0);

     QWidget *bWidget = new QWidget();
     bWidget->setLayout(bLayout);
diff --git a/Project/sources/widgets/toolbar.h
b/Project/sources/widgets/toolbar.h
index 0a31355..cd99cab 100644
--- a/Project/sources/widgets/toolbar.h
+++ b/Project/sources/widgets/toolbar.h
@@ -73,7 +73,9 @@ private:
                                 // jacob:
                                 *mScaleneObtuseTriangleButton,
                                 //JUSTIN
-
+                                //mustafa
+                                *mConvexPentagonCursorButton;
     ColorChooser *mPColorChooser, *mSColorChooser;
     bool mPrevInstrumentSetted;
     const QMap<InstrumentsEnum, QAction*> &mActMap;
(END)
```