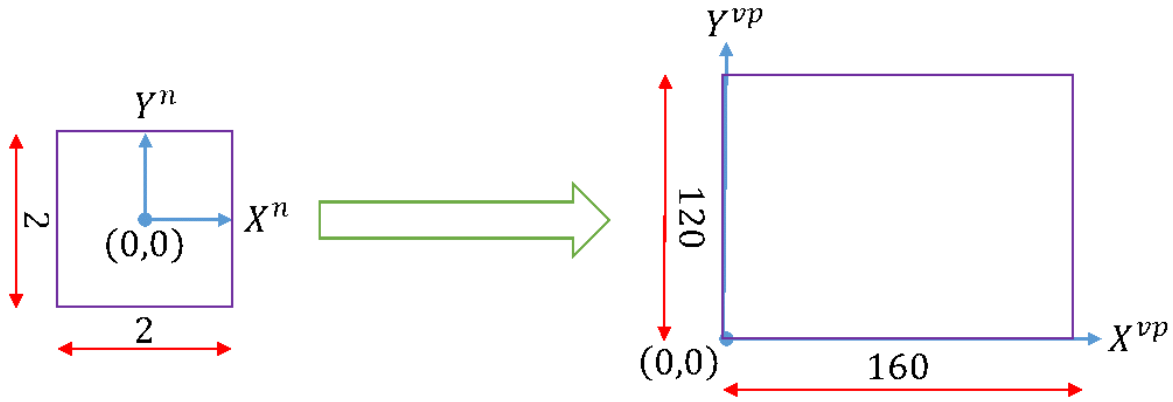
Quiz: Coordinate Systems and Viewports [Solutions]

[1] Suppose an OpenGL context window 160 pixels wide and 120 pixels high is created with a call to `glfwCreateWindow` to display rendered images.

1. Find the 3×3 homogeneous coordinates matrix that maps the 2D OpenGL NDC square to an OpenGL viewport rectangle that encompasses the entire context window.

   The following picture illustrates the mapping of the 2D OpenGL NDC square to the specified OpenGL viewport:

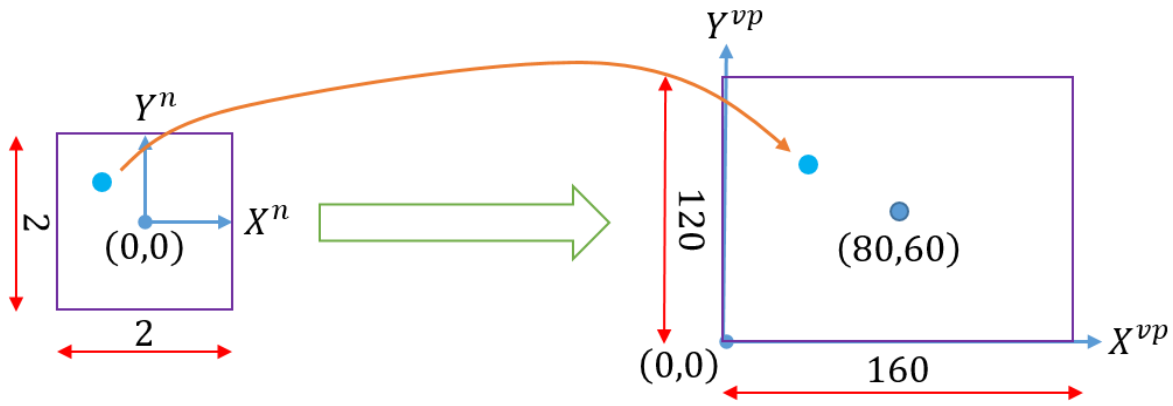


   The desired transformation matrix is:

$$\begin{bmatrix} 80 & 0 & 80 \\ 0 & 60 & 60 \\ 0 & 0 & 1 \end{bmatrix}$$

2. Find $2-$tuple pixel coordinates of a point inside the viewport corresponding to NDC point $(-0.5, 0.5)$.

   The desired mapping is illustrated below:



   Applying NDC-to-viewport transformation matrix from the previous question on NDC point $(-0.5, 0.5)$, we've:

$$\begin{bmatrix} 80 & 0 & 80 \\ 0 & 60 & 60 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -0.5 \\ 0.5 \\ 1 \end{bmatrix} = \begin{bmatrix} 40 \\ 90 \\ 1 \end{bmatrix}$$

Therefore, $\boxed{(40, 90)}$ are the pixel coordinates of a point inside the viewport corresponding to NDC point $(-0.5, 0.5)$.

3. Find $2-$tuple pixel coordinates of a point inside the viewport corresponding to NDC point $(0.1, -0.4)$.

   Applying the previously computed NDC-to-viewport transformation matrix on NDC point $(0.1, -0.4)$, we've:

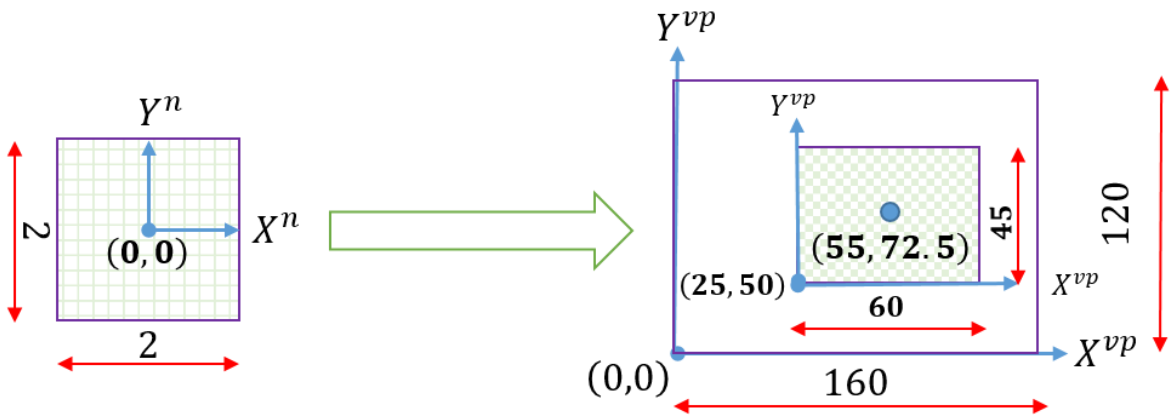
$$\begin{bmatrix} 80 & 0 & 80 \\ 0 & 60 & 60 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.1 \\ -0.4 \\ 1 \end{bmatrix} = \boxed{\begin{bmatrix} 88 \\ 36 \\ 1 \end{bmatrix}}$$

   Therefore, $\boxed{(88, 36)}$ are the pixel coordinates of a point inside the viewport corresponding to NDC point $(0.1, -0.4)$.

[2] Suppose an OpenGL context window 160 pixels wide and 120 pixels high is created with a call to `glfwCreateWindow` to display rendered images.

1. Find the 3×3 homogeneous coordinates matrix that maps the 2D OpenGL NDC square to an OpenGL viewport rectangle 60 pixels wide and 45 pixels high and whose lower-left corner has pixel coordinates $(25, 50)$. Write fractional values rounded to one decimal place.

   The following picture illustrates the mapping of the 2D OpenGL NDC square to an OpenGL viewport (the mapped regions are patterned):

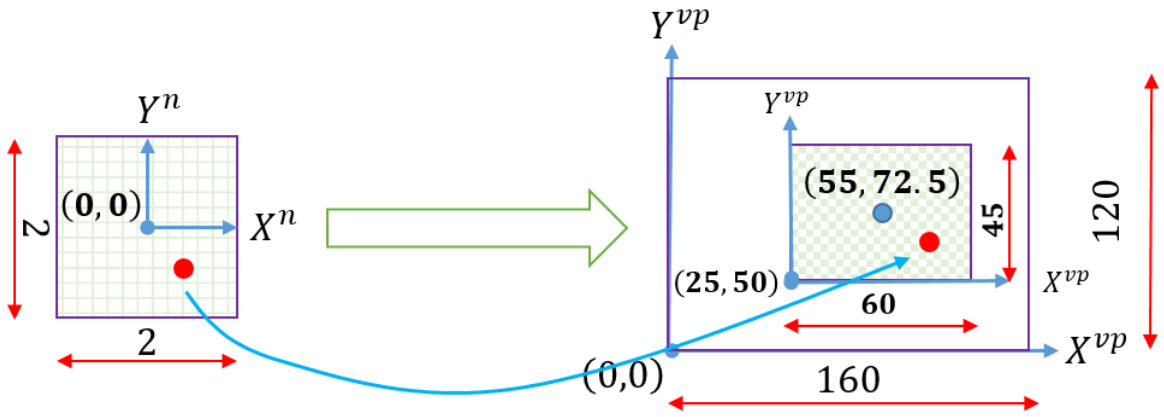


   The desired NDC-to-viewport transformation matrix is:

$$\boxed{\begin{bmatrix} 30 & 0 & 55 \\ 0 & 22.5 & 72.5 \\ 0 & 0 & 1 \end{bmatrix}}$$

2. Find $2-$tuple pixel coordinates of a point inside the viewport corresponding to NDC point $(0.5, -0.5)$. Write fractional values rounded to two decimal places.

   The desired mapping is illustrated below:



Applying the previously computed NDC-to-viewport transformation matrix on NDC point $(0.5, -0.5)$, we've:

$$\begin{bmatrix} 30 & 0 & 55 \\ 0 & 22.5 & 72.5 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.5 \\ -0.5 \\ 1 \end{bmatrix} = \boxed{\begin{bmatrix} 70 \\ 61.25 \\ 1 \end{bmatrix}}$$

Therefore, $\boxed{(70, 61.25)}$ are the pixel coordinates of a point inside the viewport corresponding to NDC point $(0.5, -0.5)$.

3. Find $2-$tuple pixel coordinates of a point inside the viewport corresponding to NDC point $(-0.1, 0.4)$. Write fractional values rounded to one decimal place.

   Applying the previously computed NDC-to-viewport transformation matrix on NDC point $(-0.1, 0.4)$, we've:

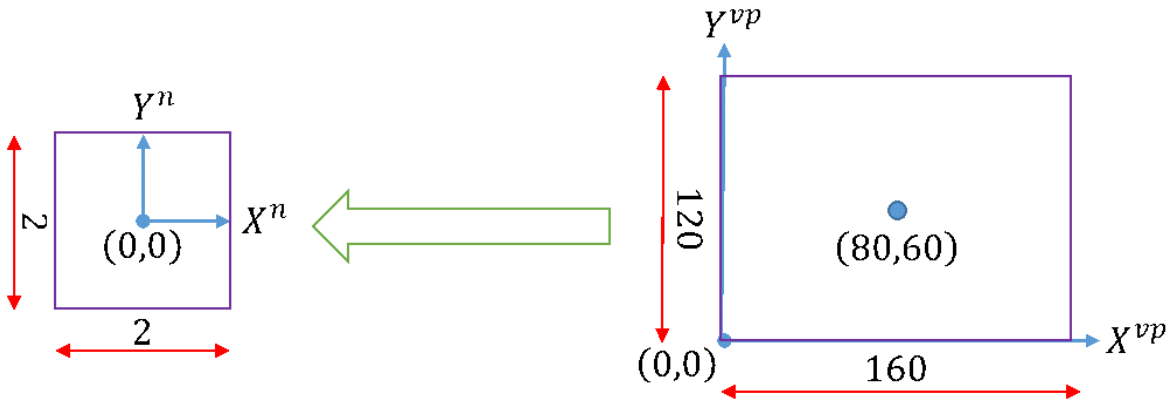
$$\begin{bmatrix} 30 & 0 & 55 \\ 0 & 22.5 & 72.5 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -0.1 \\ 0.4 \\ 1 \end{bmatrix} = \boxed{\begin{bmatrix} 52 \\ 81.5 \\ 1 \end{bmatrix}}$$

   Therefore, $\boxed{(52, 81.5)}$ are the pixel coordinates of a point inside the viewport corresponding to NDC point $(-0.1, 0.4)$.

[3] Suppose an OpenGL context window 160 pixels wide and 120 pixels high is created with a call to `glfwCreateWindow` to display rendered images.

1. If an OpenGL viewport rectangle encompasses the entire window, find the 3×3 homogeneous coordinates matrix that maps pixel coordinates of a rendered point in the viewport to the 2D OpenGL NDC square. Write fractional answers as rational values.

   The mapping of the OpenGL viewport rectangle to the NDC square is illustrated below:

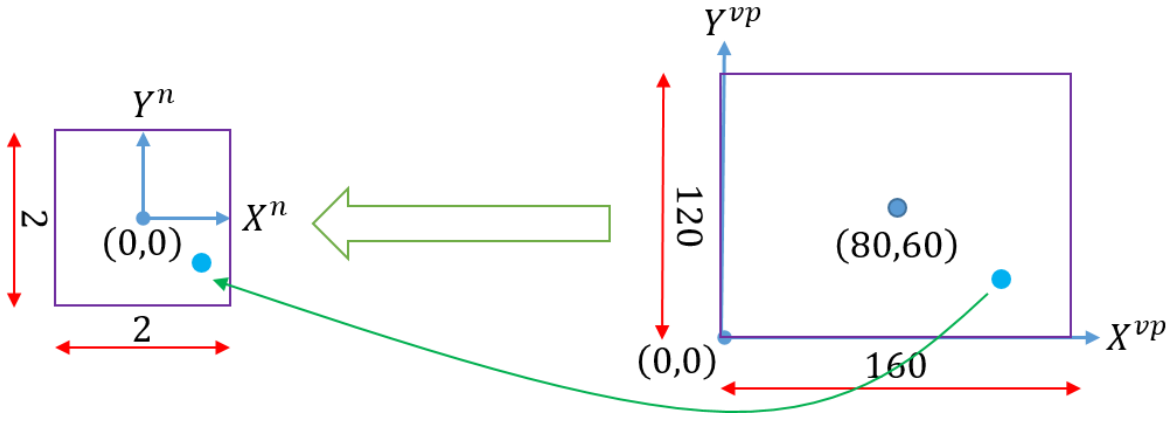


The required viewport-to-NDC transformation matrix can be computed using the exemplary method or by using the intuition that it is just the inverse of the previously computed NDC-to-viewport transformation matrix:

$$
\begin{aligned}
\mathrm{A}^{-1} &= \begin{bmatrix} 80 & 0 & 80 \\ 0 & 60 & 60 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \\
&= \left(\begin{bmatrix} 1 & 0 & 80 \\ 0 & 1 & 60 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 80 & 0 & 0 \\ 0 & 60 & 0 \\ 0 & 0 & 1 \end{bmatrix} \right)^{-1} \\
&= \begin{bmatrix} 80 & 0 & 0 \\ 0 & 60 & 0 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 1 & 0 & 80 \\ 0 & 1 & 60 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \\
&= \begin{bmatrix} \frac{1}{80} & 0 & 0 \\ 0 & \frac{1}{60} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -80 \\ 0 & 1 & -60 \\ 0 & 0 & 1 \end{bmatrix} \\
\implies \mathrm{A}^{-1} &= \boxed{\begin{bmatrix} \frac{1}{80} & 0 & -1 \\ 0 & \frac{1}{60} & -1 \\ 0 & 0 & 1 \end{bmatrix}}
\end{aligned}
$$

2. Find the $2-$tuple point inside the NDC square corresponding to the point with viewport pixel coordinates $(130, 30)$. Write fractional answers as rational values.

   The desired mapping is illustrated below:



Applying the previously computed viewport-to-NDC transformation matrix on NDC point $(130, 30)$, we've:

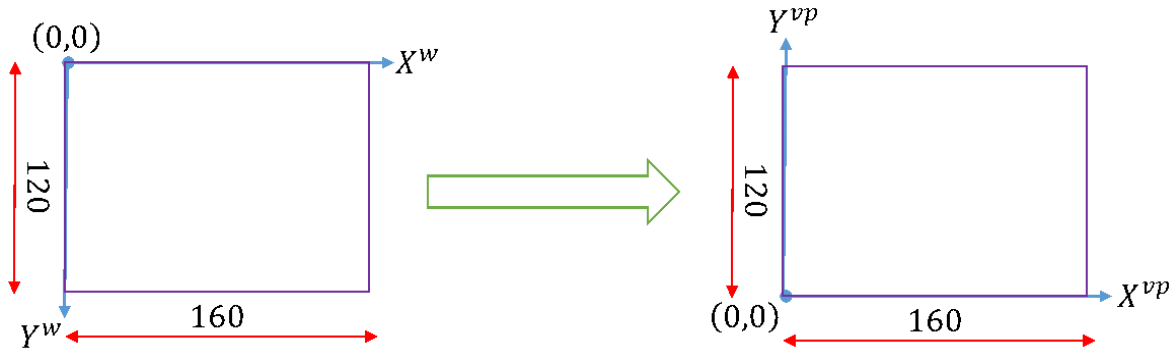
$$\begin{bmatrix} \frac{1}{80} & 0 & -1 \\ 0 & \frac{1}{60} & -1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 130 \\ 30 \\ 1 \end{bmatrix} = \boxed{\begin{bmatrix} \frac{5}{8} \\ -\frac{1}{2} \\ 1 \end{bmatrix}}$$

Therefore, $\boxed{(\frac{5}{8}, -\frac{1}{2})}$ are the pixel coordinates of a point inside the viewport corresponding to NDC point $(130, 30)$.

[4] Suppose an OpenGL context window 160 pixels wide and 120 pixels high is created with a call to `glfwCreateWindow` to display rendered images. GLFW returns the mouse cursor position inside the OpenGL context window in window coordinates, but measured relative to the upper-left corner of the window client area.

1. Find the 3×3 homogeneous coordinates matrix that maps the mouse cursor position returned by GLFW to an OpenGL viewport rectangle that encompasses the entire OpenGL context window.

   The following picture illustrates the mapping of points in the window rectangle to corresponding locations in the OpenGL viewport rectangle:

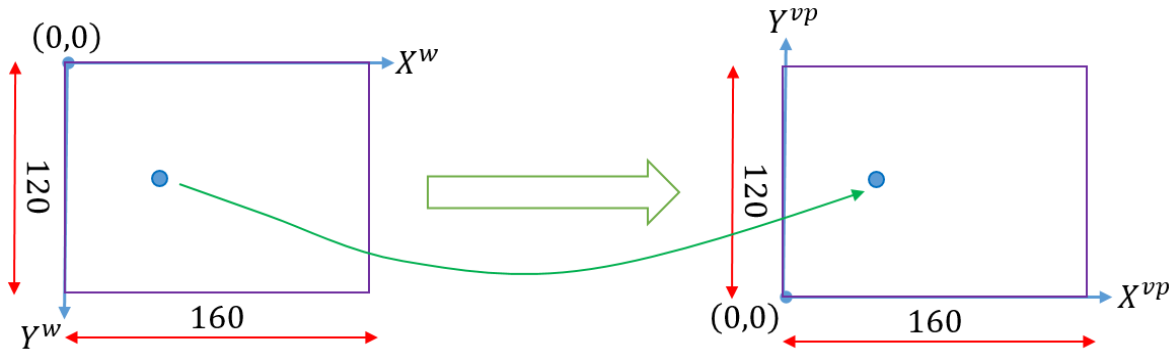


   The desired mouse-to-viewport transformation matrix is:

$$\boxed{\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 120 \\ 0 & 0 & 1 \end{bmatrix}}$$

2. If the mouse cursor position in the window client area is $(50, 60)$, find the $2-$tuple coordinates of the corresponding viewport pixel coordinates.

   The desired mapping is illustrated below:



   Applying window-to-viewport transformation matrix on window point $(50, 60)$, we've:

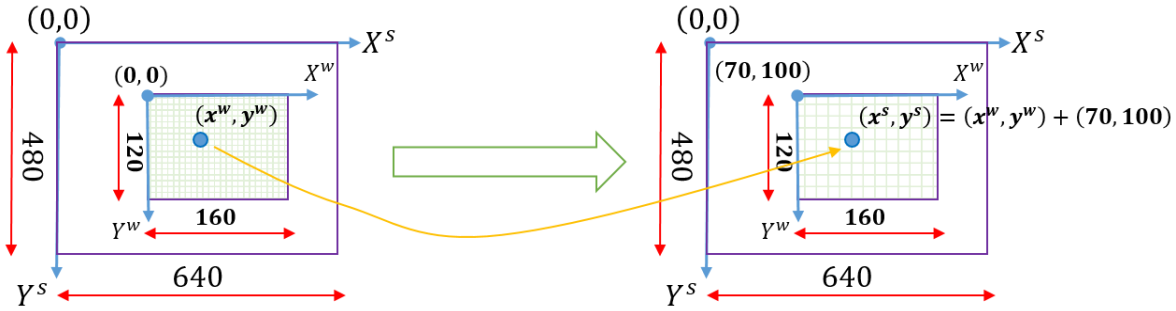
$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 120 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 50 \\ 60 \\ 1 \end{bmatrix} = \begin{bmatrix} 50 \\ 60 \\ 1 \end{bmatrix}$$

Therefore, $\boxed{(50, 60)}$ are the pixel coordinates of the mouse cursor position in the window client area is $(50, 60)$.

[5] A display device 640 pixels wide and 480 pixels high identifies pixels using screen coordinates with origin located at upper-left corner of the device, $x-$axis oriented rightward, and $y-$axis oriented downward. Consider a child window client area 160 pixel wide and 120 pixel high located at screen coordinates $(70, 100)$. The window identifies pixels using window coordinates with origin located at upper-left corner of the window, $x-$axis oriented rightward, and $y-$axis oriented downward.

1. Find the 3×3 homogeneous coordinates matrix that maps a window coordinates point in the window client area to the corresponding screen coordinates point in the display device.

   The following picture illustrates the mapping of a window coordinate point in the window client area to the corresponding screen coordinates point in the display device:

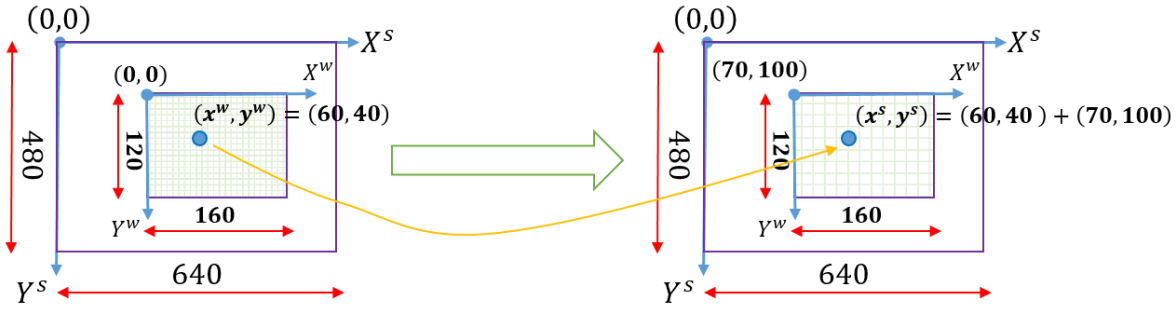


   The desired window-to-screen transformation matrix is:

$$\boxed{\begin{bmatrix} 1 & 0 & 70 \\ 0 & 1 & 100 \\ 0 & 0 & 1 \end{bmatrix}}$$

2. Find the $2-$tuple screen coordinates of a point corresponding to a window coordinates point $(60, 40)$ in the window.

   The desired mapping is illustrated below:



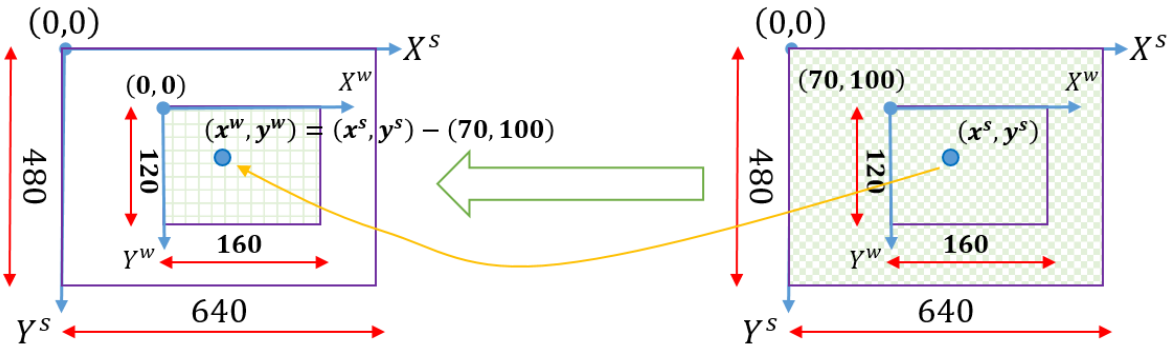
   Applying the window-to-screen transformation matrix on window point $(60, 40)$, we've:

$$\begin{bmatrix} 1 & 0 & 70 \\ 0 & 1 & 100 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 60 \\ 40 \\ 1 \end{bmatrix} = \begin{bmatrix} 130 \\ 140 \\ 1 \end{bmatrix}$$

Therefore, $\boxed{(130, 140)}$ are the screen coordinates of point corresponding to windows coordinates point $(60, 40)$ in the window.

3. Find the 3×3 homogeneous coordinates matrix that maps a screen coordinates point in the display device rectangle to the corresponding window coordinates location in the window client area.

   The following picture illustrates the mapping of a screen coordinate point in the display device to the corresponding window coordinate point in window client area:



   The required screen-to-window transformation matrix can be computed using the exemplary method or by using the intuition that it is just the inverse of the previously computed window-to-screen transformation matrix:

$$\begin{bmatrix} 1 & 0 & 70 \\ 0 & 1 & 100 \\ 0 & 0 & 1 \end{bmatrix}^{-1} = \boxed{\begin{bmatrix} 1 & 0 & -70 \\ 0 & 1 & -100 \\ 0 & 0 & 1 \end{bmatrix}}$$

4. Use the matrix from Problem 3 to find the $2-$tuple window coordinates of a point in the window client area corresponding to the screen coordinates point $(160, 120)$.

   Applying the screen-to-window transformation matrix on screen coordinates point $(160, 120)$, we've:

$$\begin{bmatrix} 1 & 0 & -70 \\ 0 & 1 & -100 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 160 \\ 120 \\ 1 \end{bmatrix} = \boxed{\begin{bmatrix} 90 \\ 20 \\ 1 \end{bmatrix}}$$

   Therefore, $\boxed{(90, 20)}$ are the window coordinates of a screen coordinates point $(160, 120)$.