



Universidad Autónoma del estado de México

Ingeniería en computación

7.mo Semestre TM

Graficación Computacional

Profesora: Hazem Álvarez Rodríguez

Instalación y ejecución de Manim & LaTeX

ALUMNO:
DIEGO ARGEL NAVARRETE GODINES

FECHA DE ENTREGA: 21 DE AGOSTO DEL 2024

Manim

Manim es un paquete para Python creado por Grant Sanderson, el host del canal de youtube 3Blue1Brown, para animar y visualizar estos mismos conceptos. Provee métodos y objetos para crear lecciones animadas sobre geometría, álgebra lineal, cálculo, física, redes neuronales, mecánica, funciones paramétricas e implícitas tanto en 2D como en 3D, y más.



Instalación de Python y Dependencias

1. Instalar Python:

Descargar Python en python.org y descarga la última versión de Python compatible con tu sistema operativo (Windows, macOS, Linux).

Ejecuta el instalador y asegúrate de seleccionar la opción "Add Python to PATH" antes de hacer clic en "Install Now". Esto configurará la variable de entorno PATH para que puedas usar Python desde la línea de comandos.



2. Verificación de Instalación de Paquetes

Para verificar que los paquetes necesarios están instalados:

pip list

```
Símbolo del sistema
Microsoft Windows [Versión 10.0.19045.4780]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\Argel>pip list
Package                Version
-----
anyio                   4.4.0
argon2-cffi             23.1.0
argon2-cffi-bindings   21.2.0
arrow                   1.3.0
asttokens               2.4.1
async-lru               2.0.4
attrs                   24.2.0
babel                   2.16.0
beautifulsoup4          4.12.3
bleach                  6.1.0
certifi                 2024.7.4
cffi                    1.17.0
charset-normalizer       3.3.2
click                   8.1.7
cloup                   3.0.5
```

3. Instalación de Manim

Si manim no se ha instalado previamente, puedes hacerlo con pip (esto también se cubrió en el paso anterior)

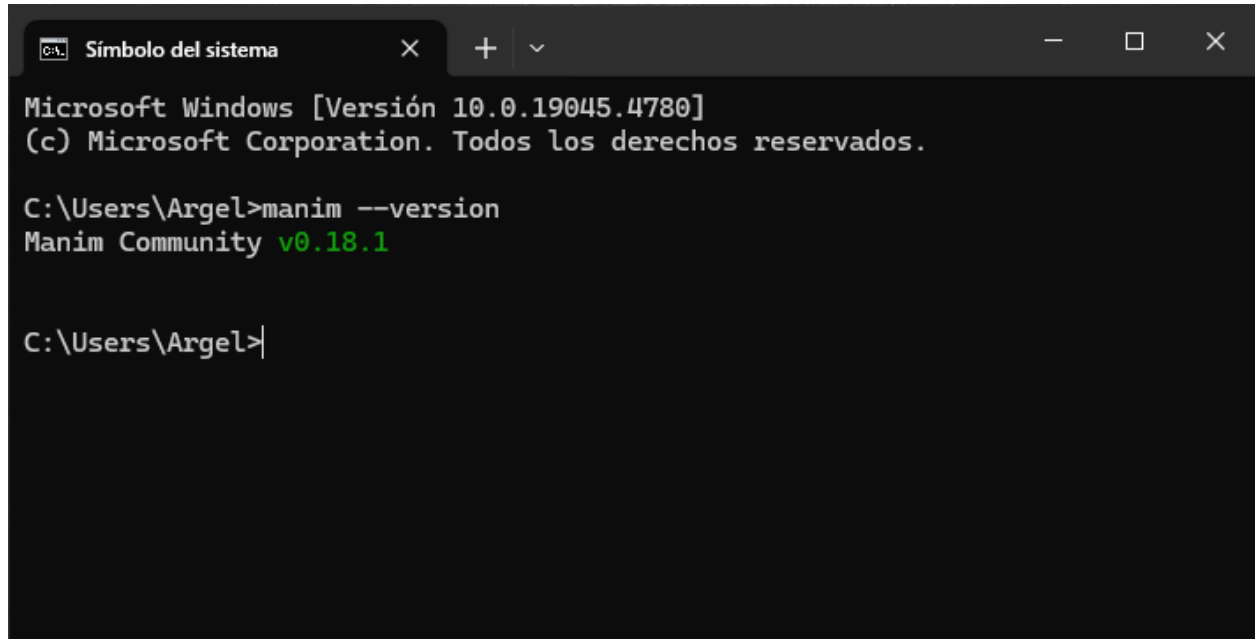
pip install manim

```
Using cached idna-3.3-py3-none-any.whl (61 kB)
Collecting certifi>=2017.4.17
  Using cached certifi-2021.10.8-py2.py3-none-any.whl (149 kB)
Collecting charset-normalizer~2.0.0
  Using cached charset-normalizer-2.0.12-py3-none-any.whl (39 kB)
Collecting urllib3<1.27,>=1.21.1
  Downloading urllib3-1.26.9-py2.py3-none-any.whl (138 kB)
    | 138 kB 3.3 MB/s
Collecting commonmark<0.10.0,>=0.9.0
  Using cached commonmark-0.9.1-py2.py3-none-any.whl (51 kB)
Collecting six
  Using cached six-1.16.0-py2.py3-none-any.whl (11 kB)
Using legacy 'setup.py install' for srt, since package 'wheel' is not installed.
Installing collected packages: six, numpy, multipledispatch, glcontext, colorama, urllib3, pyrr, Pyg
ments, pygame, Pillow, moderngl, idna, commonmark, click, charset-normalizer, certifi, watchdog, tqd
m, srt, skia-pathops, screeninfo, scipy, rich, requests, pydub, pycairo, networkx, moderngl-window,
mapbox-earcut, manimpango, isosurfaces, decorator, colour, cloup, click-default-group, backports.cac
hed-property, manim
Running setup.py install for srt ... done
Successfully installed Pillow-9.0.1 Pygments-2.11.2 backports.cached-property-1.0.1 certifi-2021.10.
8 charset-normalizer-2.0.12 click-8.0.4 click-default-group-1.2.2 cloup-0.7.1 colorama-0.4.4 colour-
0.1.5 commonmark-0.9.1 decorator-5.1.1 glcontext-2.3.5 idna-3.3 isosurfaces-0.1.0 manim-0.15.1 manim
pango-0.4.0.post2 mapbox-earcut-0.12.11 moderngl-5.6.4 moderngl-window-2.4.1 multipledispatch-0.6.0
networkx-2.7.1 numpy-1.22.3 pycairo-1.21.0 pydub-0.25.1 pygame-1.5.2 pyrr-0.10.3 requests-2.27.1 ri
ch-12.0.0 scipy-1.8.0 screeninfo-0.8 six-1.16.0 skia-pathops-0.7.2 srt-3.5.2 tqdm-4.63.0 urllib3-1.2
6.9 watchdog-2.1.6
WARNING: You are using pip version 21.2.3; however, version 22.0.4 is available.
You should consider upgrading via the 'C:\dev\manim-project\.venv\Scripts\python.exe -m pip install
--upgrade pip' command.
(.venv) PS C:\dev\manim-project>
```

4. Verificación de Instalación de Manim

Para verificar que manim está instalado correctamente, ejecuta:

manim --version



```
Símbolo del sistema
Microsoft Windows [Versión 10.0.19045.4780]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\Argel>manim --version
Manim Community v0.18.1

C:\Users\Argel>
```

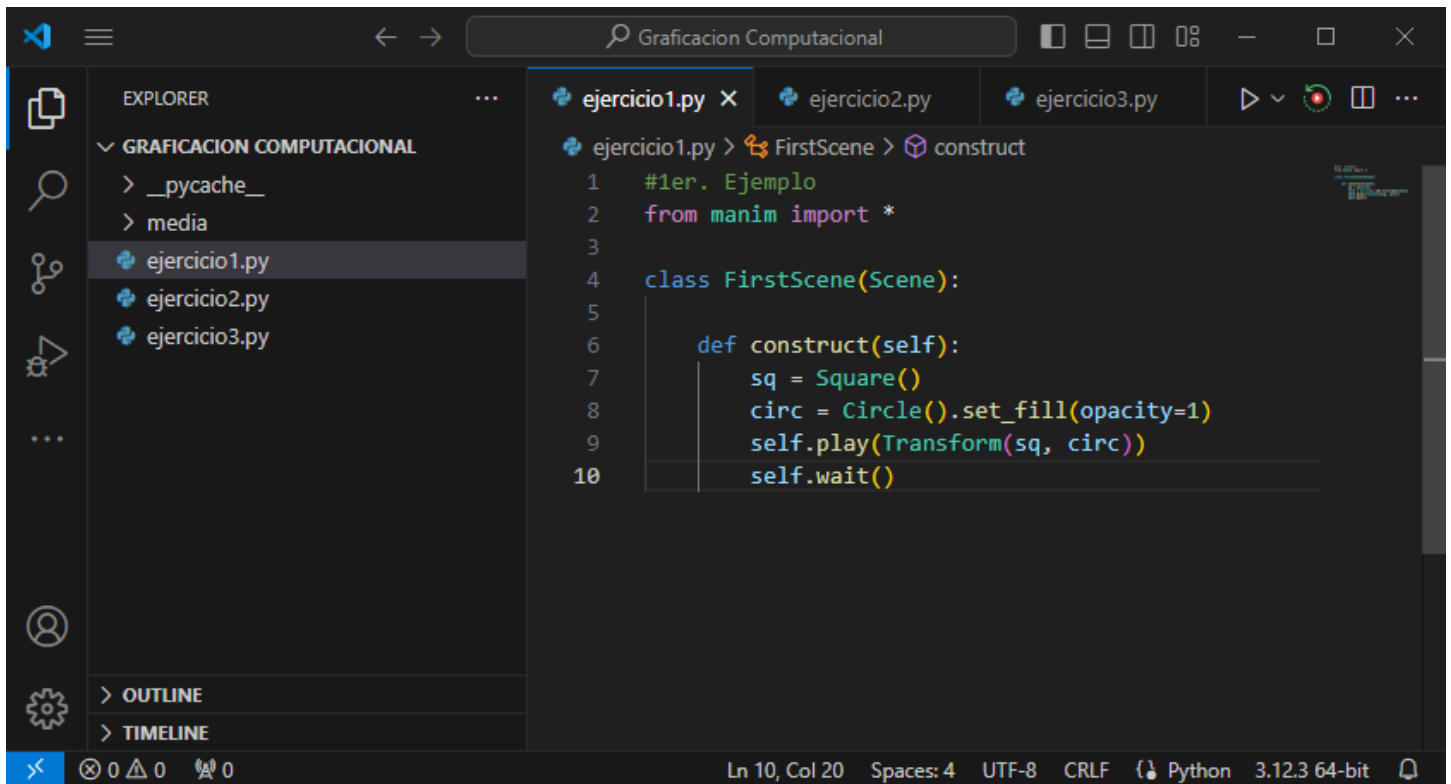
5. Trabajar con Ejemplos en Visual Studio Code

Configuración de Visual Studio Code:

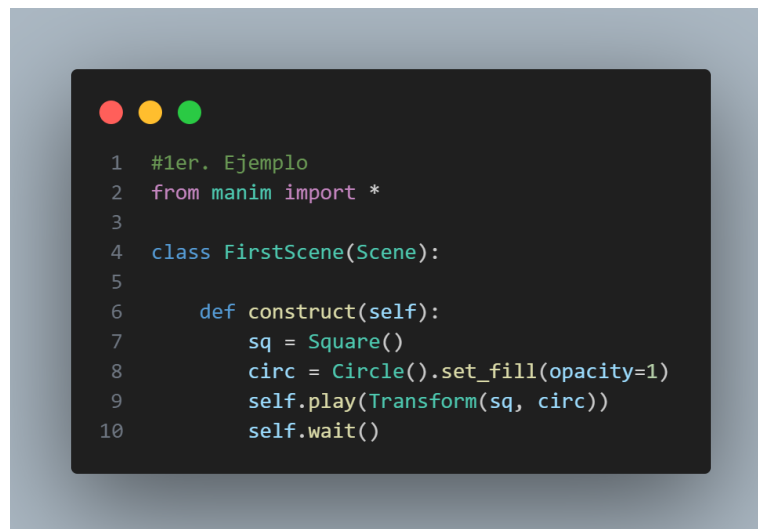
- Instala Visual Studio Code: Descárgalo e instálalo desde code.visualstudio.com.
- Instala la extensión de Python: Abre Visual Studio Code, ve a la pestaña de Extensiones (Ctrl+Shift+X), y busca e instala la extensión de Python proporcionada por Microsoft.

Crear un nuevo proyecto:

- Abre Visual Studio Code y abre la carpeta donde quieres guardar tus archivos de proyecto.
- Crea un archivo nuevo para cada ejemplo, por ejemplo ejemplo1.py, ejemplo2.py, etc.



Escribir código en manim en Python Ejemplo1_manim:



Este script de Manim crea una animación básica donde un cuadrado se transforma en un círculo, se importa todo lo necesario de la biblioteca Manim y se define una clase FirstScene que hereda de Scene, la cual representa la escena de la animación, dentro del método construct, que es donde se define el contenido de la animación, se crea un cuadrado (Square()) y un círculo (Circle()) con opacidad completa, la función self.play se usa para animar la transformación del cuadrado en el círculo y para terminar el self.wait() hace que la animación se detenga y se quede en pantalla unos segundos antes de finalizar.

Para poder ejecutar cada uno de los ejemplos es usando la siguiente sintaxis:

manim Archivo.py NombreClase -p

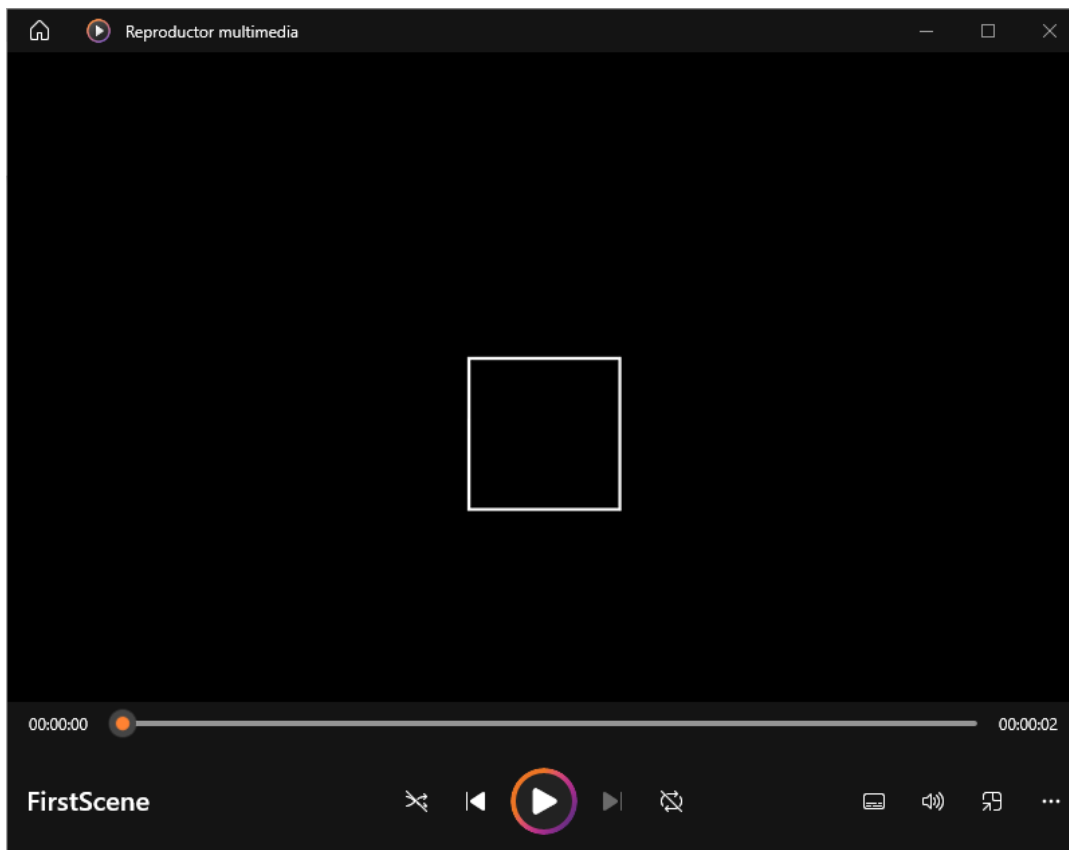
```
Símbolo del sistema
C:\Users\Argel\Desktop\Graficacion Computacional>manim ejercicio1.py FirstScene -p
Manim Community v0.18.1

[08/21/24 22:09:19] INFO Animation 0 : Using cached data (hash : 3977891868_4016253_3256495558) cairo_renderer.py:88
INFO Animation 1 : Using cached data (hash : 2852726489_1704852926_124105690) cairo_renderer.py:88
INFO Combining to Movie file. scene_file_writer.py:617
[08/21/24 22:09:21] INFO File ready at 'C:\Users\Argel\Desktop\Graficacion
Computacional\media\videos\ejercicio1\1080p60\FirstScene.mp4' scene_file_writer.py:737

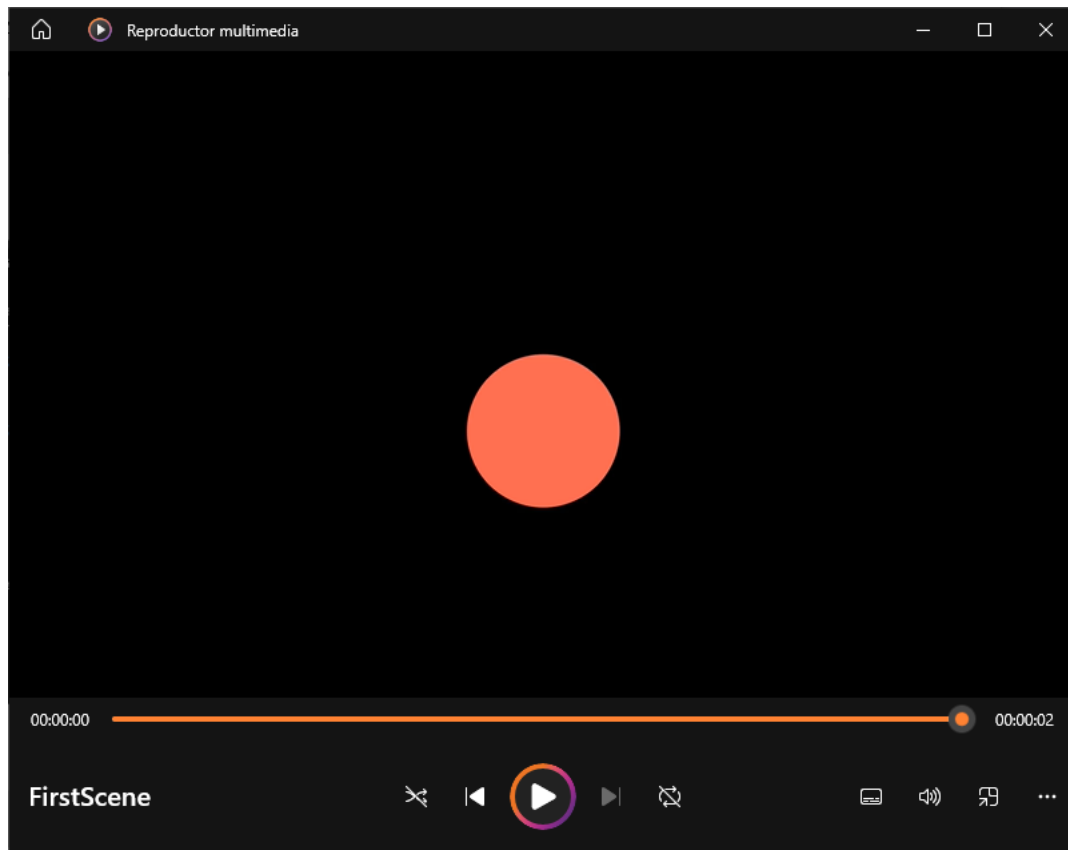
INFO Rendered FirstScene scene.py:247
INFO Played 2 animations
INFO Previewed File at: 'C:\Users\Argel\Desktop\Graficacion
Computacional\media\videos\ejercicio1\1080p60\FirstScene.mp4' file_ops.py:231

C:\Users\Argel\Desktop\Graficacion Computacional>
```

Obtenemos como resultado lo siguiente:



Para posteriormente se termine así



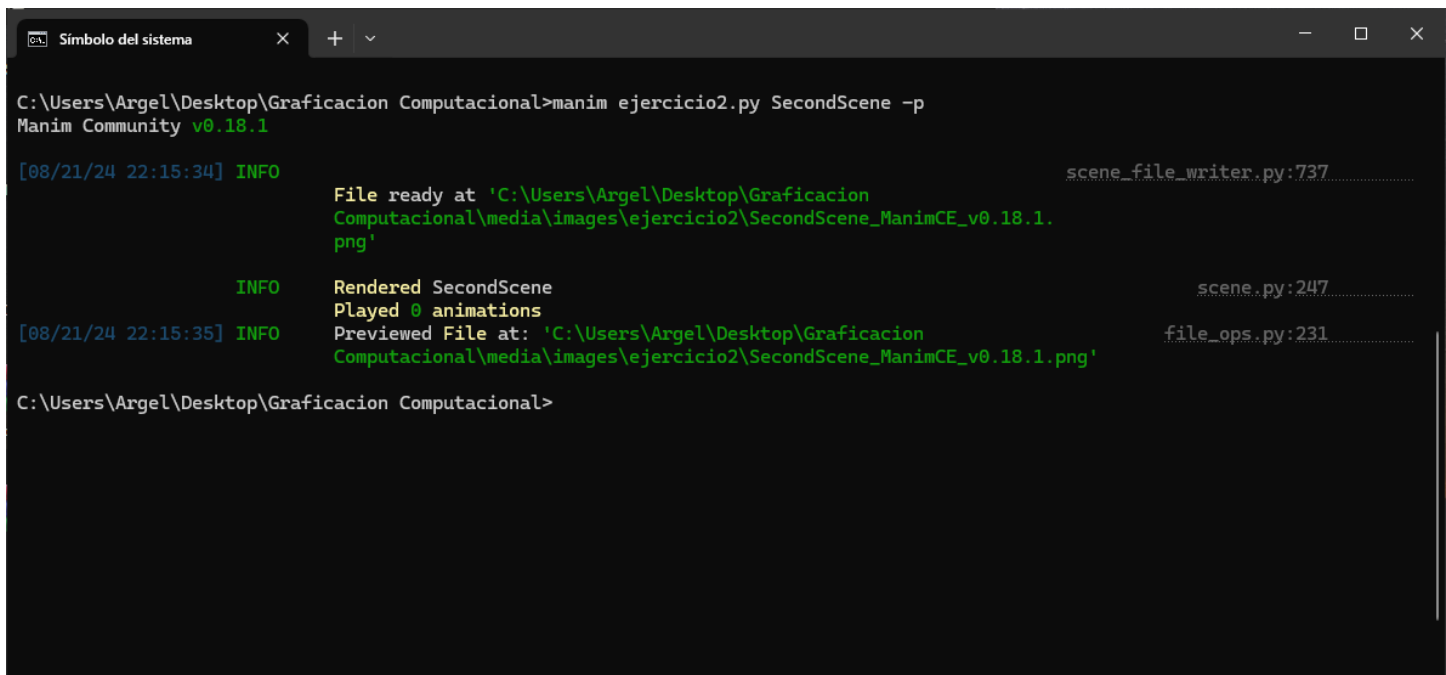
Escribir código en manim en Python Ejemplo2_LateX:

```
1 from manim import *
2
3 class SecondScene(Scene):
4     def construct(self):
5         text = MathTex("x^2")
6         self.add(text)
```

Este script de Manim crea una escena que muestra una expresión matemática en pantalla ya que se importa todo lo necesario de la biblioteca Manim, se define una clase SecondScene que hereda de Scene, representando la escena de la animación, dentro del método construct, se crea un objeto MathTex que contiene la expresión matemática "x^2", para poder mostrar se “self.add(text)” añade este texto a la escena para que se muestre cuando se renderiza la animación.

Para poder ejecutar cada uno de los ejemplos es usando la siguiente sintaxis:

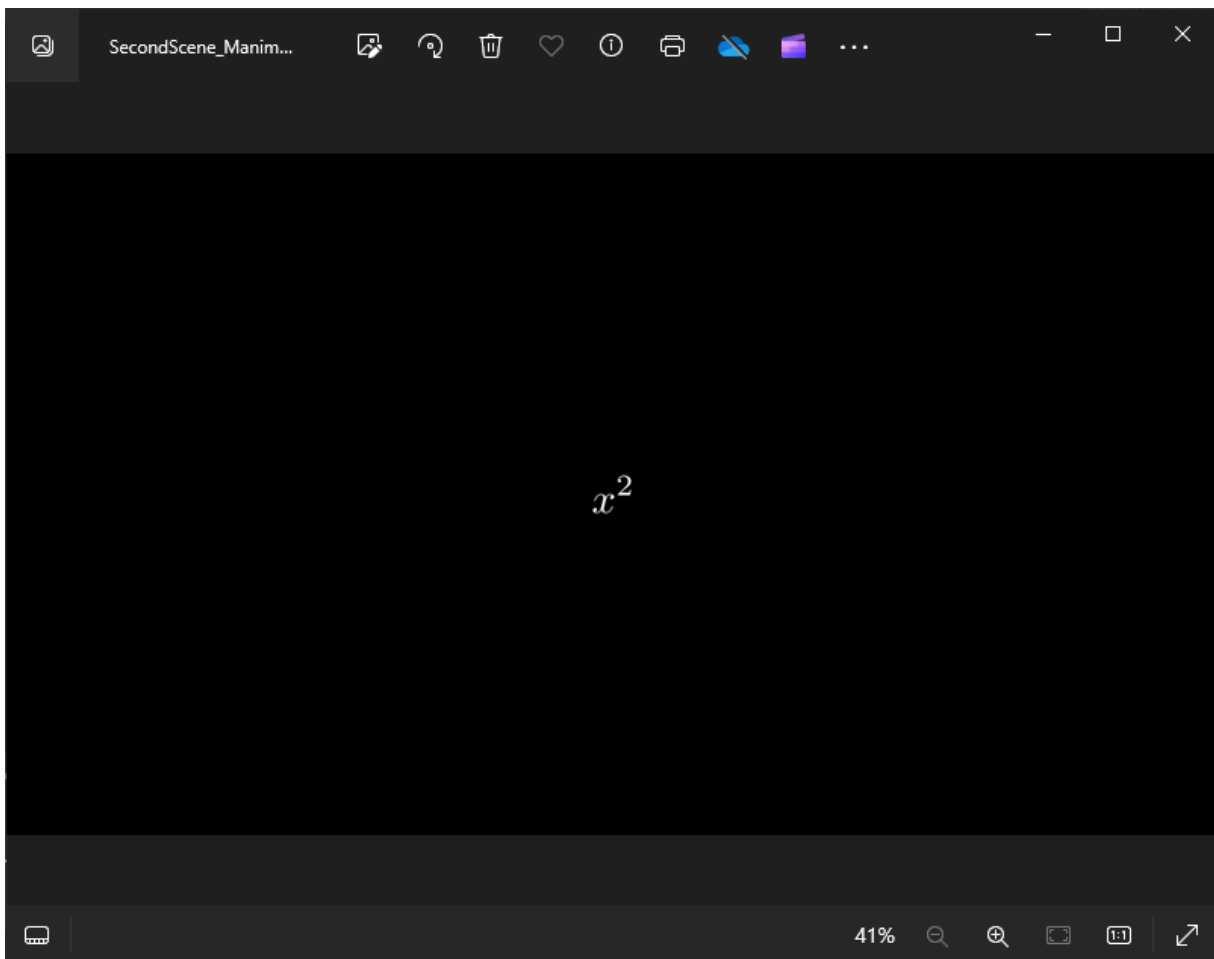
manim Archivo.py NombreClase -p



```
Símbolo del sistema
C:\Users\Argel\Desktop\Graficacion Computacional>manim ejercicio2.py SecondScene -p
Manim Community v0.18.1

[08/21/24 22:15:34] INFO File ready at 'C:\Users\Argel\Desktop\Graficacion Computacional\media\images\ejercicio2\SecondScene_ManimCE_v0.18.1.png' scene_file_writer.py:737
INFO Rendered SecondScene scene.py:247
[08/21/24 22:15:35] INFO Played 0 animations
INFO Previewed File at: 'C:\Users\Argel\Desktop\Graficacion Computacional\media\images\ejercicio2\SecondScene_ManimCE_v0.18.1.png' file_ops.py:231
C:\Users\Argel\Desktop\Graficacion Computacional>
```

Obtenemos como resultado lo siguiente:



Escribir código en manim en Python Ejemplo3_FormulaGeneral:

```
1 from manim import *
2
3 class ThreeScene(Scene):
4     def construct(self):
5         text = MathTex(r"X = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}")
6         self.add(text)
```

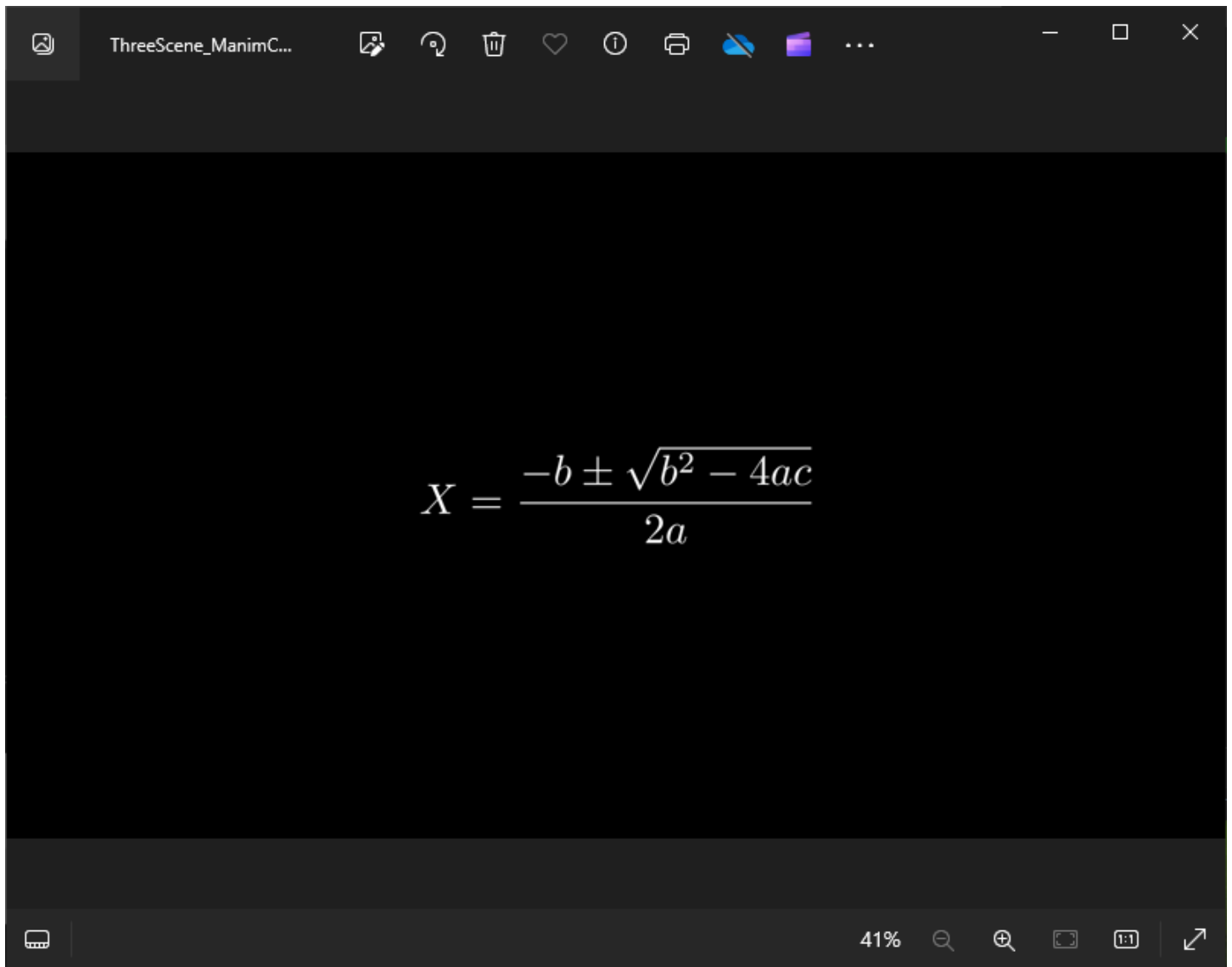
Este script de Manim crea una escena que muestra una fórmula matemática en pantalla, se importa la biblioteca Manim y se define la clase ThreeScene, que hereda de Scene, para representar la animación, este método construct, se crea un objeto MathTex con la fórmula cuadrática en formato LaTeX: $X = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$. La función self.add(text) añade esta fórmula a la escena, haciendo que aparezca en la animación cuando se renderiza.

Para poder ejecutar cada uno de los ejemplos es usando la siguiente sintaxis:

manim Archivo.py NombreClase -p

```
Símbolo del sistema x + v - □ x
INFO Rendered SecondScene scene.py:247
[08/21/24 22:15:35] INFO Played 0 animations
Previewed File at: 'C:\Users\Argel\Desktop\Graficacion Computacional\media\images\ejercicio2\SecondScene_ManimCE_v0.18.1.png' file_ops.py:231
C:\Users\Argel\Desktop\Graficacion Computacional>manim ejercicio3.py ThreeScene -p
Manim Community v0.18.1
[08/21/24 22:22:56] INFO scene_file_writer.py:737
File ready at 'C:\Users\Argel\Desktop\Graficacion Computacional\media\images\ejercicio3\ThreeScene_ManimCE_v0.18.1.png'
INFO Rendered ThreeScene scene.py:247
[08/21/24 22:22:57] INFO Played 0 animations
Previewed File at: 'C:\Users\Argel\Desktop\Graficacion Computacional\media\images\ejercicio3\ThreeScene_ManimCE_v0.18.1.png' file_ops.py:231
C:\Users\Argel\Desktop\Graficacion Computacional>
```

Obtenemos como resultado lo siguiente:


$$X = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

GitHub

<https://github.com/DiegoNavarrete05/GrafComp24B>

Conclusión:

Los ejercicios en Manim, como los ejemplos de código, ilustran cómo usar esta potente biblioteca para crear animaciones matemáticas y científicas de manera efectiva, cada ejercicio muestra diferentes aspectos de Manim, desde la transformación de formas geométricas hasta la visualización de fórmulas matemáticas complejas, aprender a utilizar Manim implica familiarizarse con la creación de objetos gráficos, la aplicación de animaciones y la personalización de presentaciones visuales, a medida que se exploran y combinan estos conceptos, es posible crear visualizaciones dinámicas y educativas que faciliten la comprensión de conceptos matemáticos y científicos.