UNIVERSIDAD DE GUADALAJARA

Centro Universitario de Ciencias Exactas e ingenierías

Computación Tolerante a Fallas

Estatus

Nombre: Castillo Mares Gilberto

Código: 213330514

Profesor: Dr. Michel Emanuel López Franco

2024-B

Desarrollo

Realizar un programa que sea capaz de revisar el estado de tu aplicación.

Para esto vamos a crear un servicio para Windows el cual, en este caso, monitoreará el programa que anteriormente hice, un juego de adivinanzas. Este servicio se ejecutará cada vez que se inicie el sistema operativo y siempre estará en ejecución, revisará si el programa esta cerrado o abierto, dependiendo la situación hará lo contrario, por ejemplo, si el programa está cerrado lo abrirá en segundo plano, y si ya esta abierto en primer plano, lo cerrará.

Primero necesitaremos usar Python para crear el script, a su vez se necesitará instalar el módulo ***psutil.***

Este modulo se puede instalar desde la consola de comandos de Windows (cmd) o PowerShell con el siguiente comando:

```
PS F:\2024A\Estatus> pip install psutil
```

Siguiendo con el código del script en Python:

```
Servicio.py > ...
1   import subprocess
2   import time
3   import psutil
4
```

1. **import subprocess**: Este módulo permite a Python crear y controlar procesos secundarios (subprocesos). Se utiliza para iniciar un nuevo proceso, en este caso, el proceso que se desea monitorear.
2. **import time**: Este módulo proporciona funciones relacionadas con la manipulación del tiempo. Se utiliza para hacer que el script espere un cierto tiempo antes de la próxima iteración del bucle.

3. **import psutil**: Este módulo proporciona una interfaz para trabajar con procesos y el sistema operativo. Se utiliza para obtener información sobre los procesos en ejecución.

```
4
5   def checar_Proceso(nombre_Proceso):
6       for proceso in psutil.process_iter(['pid', 'name']):
7           if proceso.info['name'] == nombre_Proceso:
8               return True
9       return False
```

def checar_Proceso(nombre_Proceso): Esta función toma el nombre de un proceso como argumento y devuelve **True** si el proceso está en ejecución y **False** si no lo está. Utiliza la biblioteca **psutil** para obtener información sobre los procesos en ejecución y compara el nombre del proceso con el nombre proporcionado.

```
11  def iniciar_Proceso(ruta_Proceso):
12      subprocess.Popen(ruta_Proceso)
```

def iniciar_Proceso(ruta_Proceso): Esta función toma la ruta de un proceso ejecutable como argumento y lo inicia utilizando el módulo **subprocess**. Utiliza **subprocess.Popen()** para iniciar el proceso.

```
14  def parar_Proceso(nombre_Proceso):
15      for proceso in psutil.process_iter(['pid', 'name']):
16          if proceso.info['name'] == nombre_Proceso:
17              proceso.kill()
```

def parar_Proceso(nombre_Proceso): Esta función toma el nombre de un proceso como argumento y lo detiene utilizando el módulo **psutil**. Utiliza **psutil.Process().kill()** para detener el proceso.

```
9   if __name__ == '__main__':
0       nombre_Proceso = 'Adivinanza.exe'
1       ruta_Proceso = r'F:\2024A\Estatus\dist\Adivinanza.exe'
2
```

1. **if __name__ == '__main__':**: Esta línea comprueba si el script se está ejecutando como un programa principal (no como un módulo importado). Si es así, ejecuta el siguiente código.
2. **nombre_Proceso = 'Adivinanza.exe'**: Esto establece el nombre del proceso que se va a monitorear.
3. **ruta_Proceso = r'F:\2024A\Estatus\dist\Adivinanza.exe'**: Esto establece la ruta del ejecutable del proceso que se va a iniciar. Nota: debe ser un ejecutable con extensión .exe, en este caso porque estamos trabajando con Windows)

```
while True:
    if not checar_Proceso(nombre_Proceso):
        iniciar_Proceso(ruta_Proceso)
    else:
        parar_Proceso(nombre_Proceso)

    time.sleep(5)
```

1. **while True:**: Este bucle se ejecuta indefinidamente.
2. **if not checar_Proceso(nombre_Proceso):**: Si el proceso no está en ejecución, se inicia utilizando **iniciar_Proceso(ruta_Proceso)**.
3. **else:**: Si el proceso está en ejecución, se detiene utilizando **parar_Proceso(nombre_Proceso)**.
4. **time.sleep(5)**: El script espera 5 segundos antes de la próxima iteración del bucle. Esto se hace para evitar que el script se ejecute demasiado rápido y consuma recursos innecesarios.

Hasta aquí hemos creado un script que se ejecutará cada cierto tiempo para monitorear el programa indicado en la ruta, en este caso es un pequeño juego de adivinanzas.

Después de esto necesitaremos usar un pequeño software de código libre, *NSSM Service Installer* el cual nos ayudara a crear un servicio para Windows.

En esta misma carpeta se encuentra el software, al igual que el script y esta pequeña guía.

Este software se debe encontrar en el mismo directorio que el script, así que lo ejecutaremos usando la consola de PowerShell.

Para esto debemos ingresar directamente al directorio donde se encuentran los archivos.

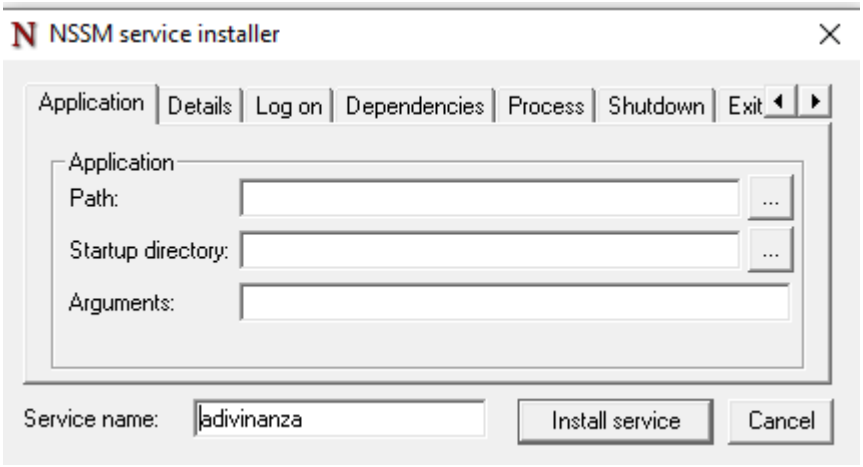
```
PS C:\Users\Gilberto> cd F:\2024A\Estatus
PS F:\2024A\Estatus>
```

Una vez en el directorio, ingresamos el nombre del software seguido de un comando:

```
PS F:\2024A\Estatus> .\nssm.exe install adivinanza
```

Ingresamos el comando **install** y después el nombre que nosotros queramos que se llame el servicio.

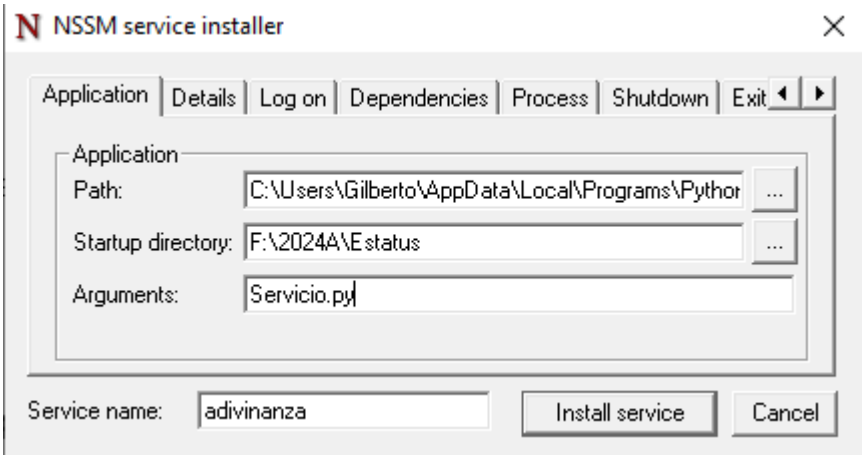
Después nos aparecerá una pequeña ventana:



En **Path**, debemos colocar la ruta donde se encuentra el ejecutable de Python.

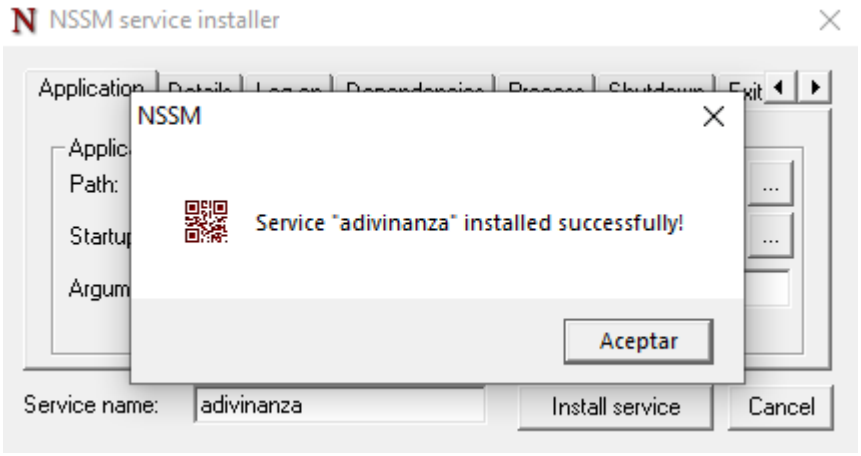
En **Startup directory**, debemos colocar la ruta del directorio donde se encuentra nuestro script de Python.

En **Arguments**, ya que el script solo se debe ejecutar “invocando” su nombre, solo colocaremos el nombre que le dimos al script.



Después le damos al botón de **Install service** para instalar nuestro servicio.

Aparecerá un pequeño mensaje en el software, así como en la consola:




```
Service "adivinanza" installed successfully!
```

Luego de esto debemos iniciar nuestro servicio con el siguiente comando en PowerShell:

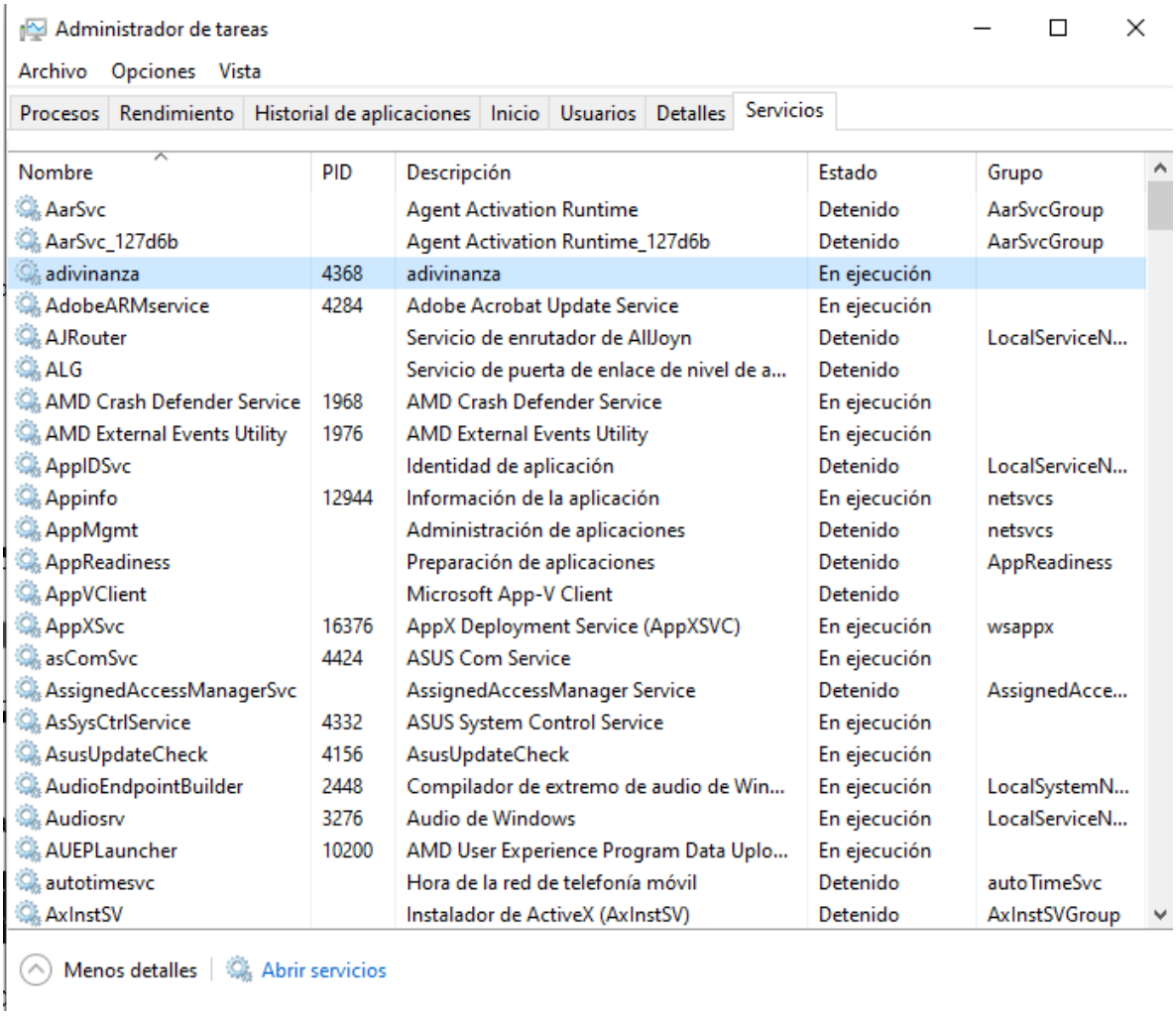
```
PS F:\2024A\Estatus> .\nssm.exe start adivinanza
```

Debemos colocar el nombre del software, así como la palabra **start** y el nombre que le dimos a nuestro servicio.

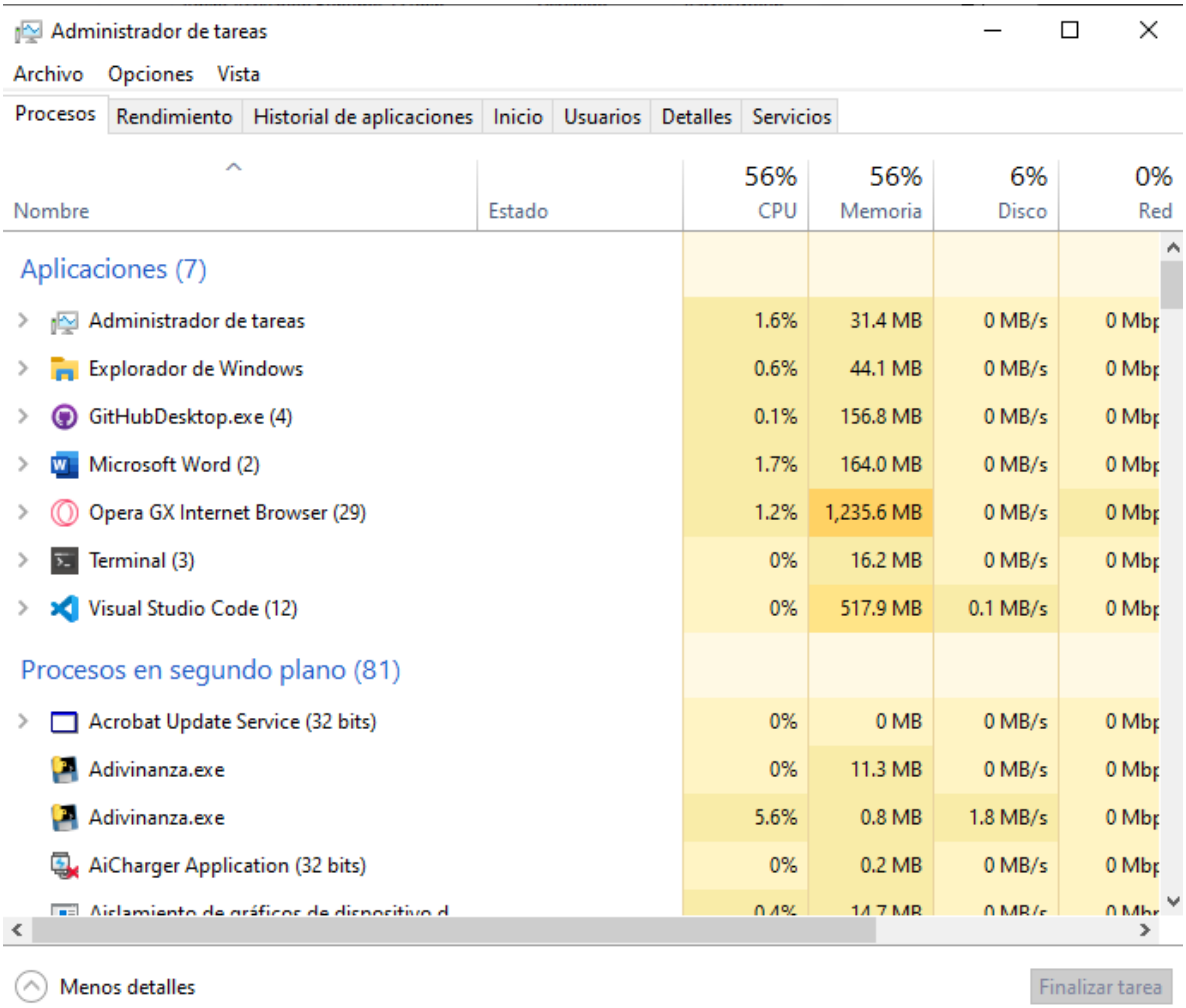
Después no aparecerá este mensaje en la consola:

```
adivinanza: START: La operación se completó correctamente.
PS F:\2024A\Estatus>
```

Habiendo hecho esto, vamos al Administrador de tareas de Windows en la pestaña **Servicios** para verificar que el servicio se instaló y se está ejecutando:



Vamos a la pestaña de **Procesos** para verificar el estado de nuestro programa:



En la parte de procesos en segundo plano podemos ver que ahí se encuentra nuestro programa ejecutándose. Debido a que en el script está colocado con una extensión de tiempo de 5 segundos aproximadamente, después de este tiempo el servicio detiene el proceso y lo cierra.

Si ejecutas manualmente el programa, después de 5 segundos éste se cerrará y después de otros 5 segundos se ejecutará en segundo plano.

Si necesitas parar el servicio, puedes usar el siguiente comando:

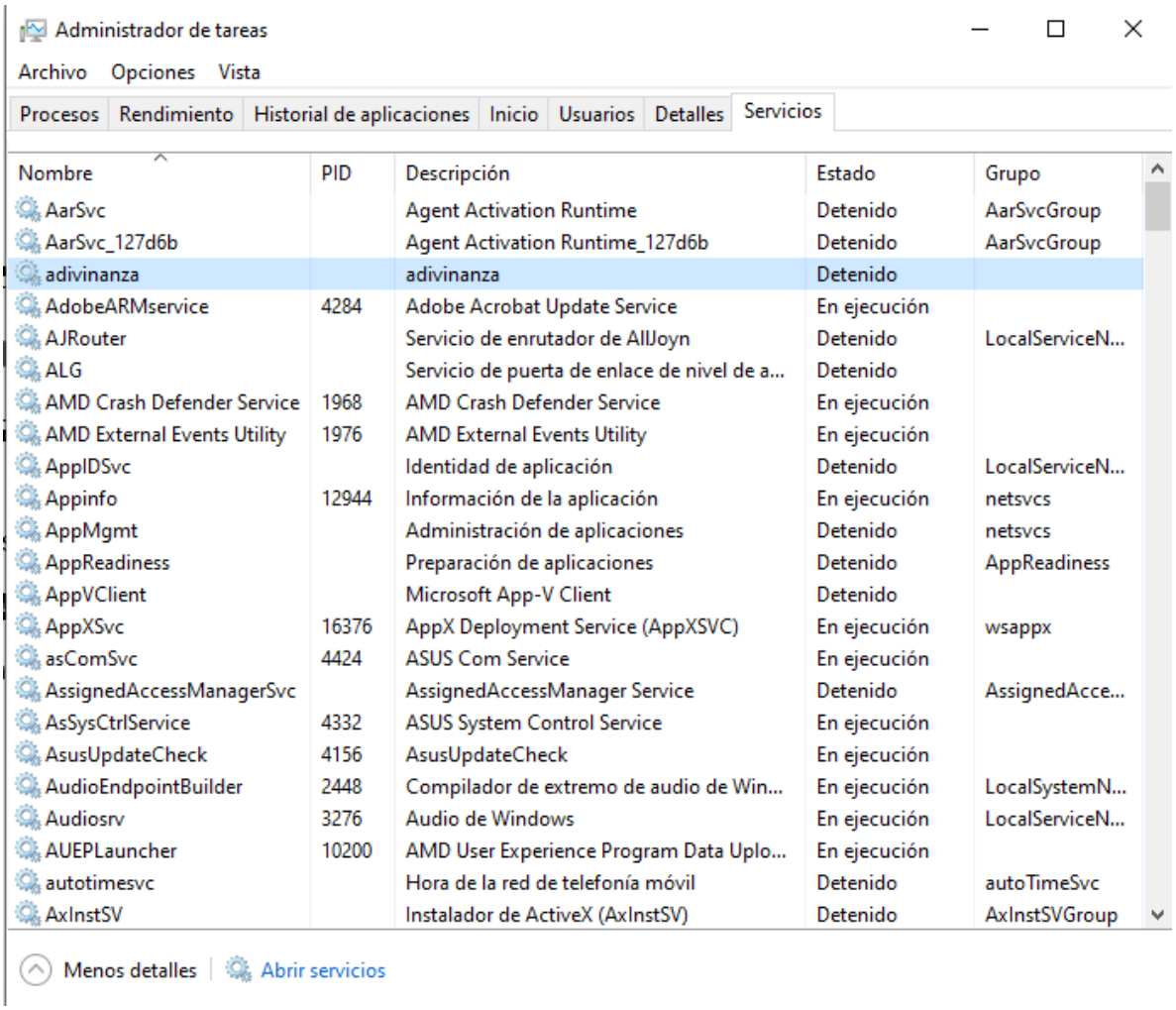
```
PS F:\2024A\Estatus> .\nssm.exe stop adivinanza
```

Aquí colocamos el nombre del software, seguido de la palabra **stop** y el nombre del servicio.

Aparecerá el siguiente mensaje:

```
adivinanza: STOP: La operación se completó correctamente.
```

Podemos confirmar esto en el Administrador de tareas:



De la misma forma podemos editar nuestro servicio usando el siguiente comando:

```
PS F:\2024A\Estatus> .\nssm.exe edit adivinanza
```

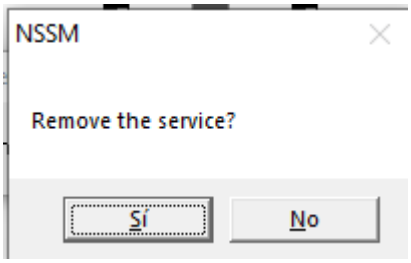
Nos volverá a aparecer la pequeña ventana donde podremos editar el servicio, **NOTA: para hacer cualquier edición al servicio, siempre se tiene que detener el servicio primero y luego editarlo.**

Aunque la computadora se apague, el servicio se quedará, cuando se encienda de nuevo, el servicio se ejecutará sin que el usuario intervenga.

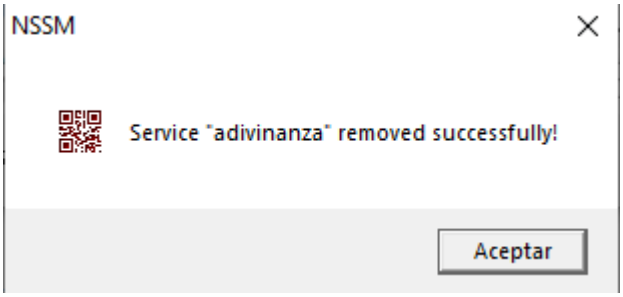
Por último, si deseas remover el servicio puedes hacerlo con el siguiente comando (recuerda que debes detener el servicio antes):

```
PS F:\2024A\Estatus> .\nssm.exe remove adivinanza
```

El programa preguntará por ultima vez si estás seguro



Después aparecerá un mensaje



También en consola

```
Service "adivinanza" removed successfully!
```

Así podremos crear nuestros servicios para Windows usando scripts de Python.

Código completo:

```python
import subprocess
import time
import psutil

def checar_Proceso(nombre_Proceso):
    for proceso in psutil.process_iter(['pid', 'name']):
        if proceso.info['name'] == nombre_Proceso:
            return True
    return False

def iniciar_Proceso(ruta_Proceso):
    subprocess.Popen(ruta_Proceso)

def parar_Proceso(nombre_Proceso):
    for proceso in psutil.process_iter(['pid', 'name']):
        if proceso.info['name'] == nombre_Proceso:
            proceso.kill()

if __name__ == '__main__':
    nombre_Proceso = 'Adivinanza.exe'
    ruta_Proceso = r'F:\2024A\Estatus\dist\Adivinanza.exe'

    while True:
        if not checar_Proceso(nombre_Proceso):
            iniciar_Proceso(ruta_Proceso)
        else:
            parar_Proceso(nombre_Proceso)

        time.sleep(5)
```

Bibliografía

tecnobillo. (s. f.). *Crear Servicios para Windows con Python*. Tecnobillo. https://tecnobillo.com/sections/python-en-windows/servicios-windows-python/servicios-windows-python.html

NSSM - the Non-Sucking Service Manager. (s. f.). https://nssm.cc/usage

NSSM - the Non-Sucking Service Manager. (s. f.-b). https://nssm.cc/download

Error : No module named «psutil». (s. f.). Stack Overflow. https://stackoverflow.com/questions/50316358/error-no-module-named-psutil

Threading: programación con hilos (I). (s. f.). https://python-para-impacientes.blogspot.com/2016/12/threading-programacion-con-hilos-i.html