Tabla de Contenidos

[Introducción 2](#_Toc480147649)

[Descripción del problema: 2](#_Toc480147650)

[Dificultades encontradas: 2](#_Toc480147651)

[Análisis de Resultados: 4](#_Toc480147652)

[Bitácora de actividades: 4](#_Toc480147653)

[Estadística de tiempos: 5](#_Toc480147654)

[Conclusión Personal 5](#_Toc480147655)

# Introducción

El objetivo de este proyecto programado es realizar un robot y una ventana de comandos para que el robot realice ciertas acciones ya predefinidas. Además, habrá que recuperar el estado del robot y administrar el robot por medio de “energía”. Cabe destacar que las funciones por realizar se encuentran en la pantalla de ayuda del programa.

# Descripción del problema:

El problema principal se encuentra en la elaboración de las acciones del robot ya que equivale a la mayor parte del problema y de ahí surgen problemas “secundarios”. De ahí surgen

# Dificultades encontradas:

1. Diseño del robot: Para crear mi robot después de investigar y llegar a la conclusión que, debo yo dibujar mi robot, para poder obtener los *frames* de las animaciones. Así, investigue sobre “vector graphics” y su utilidad para diseños digitales, entonces usando Sketch (un programa para diseñar alternativo a Adobe Illustrator) logre diseñar mi robot y obtener diferentes imágenes en diferentes posiciones para crear las animaciones.
2. Animaciones: El desafío de mostrar animaciones ha sido el más largo y se trató de diferentes formas. Al inicio se trató de mostrar un archivo con formato GIF que ya estuviera animado, debido a que investigando aprendí que a la hora de abrir el archivo con tkinter hay una opción de formato para que muestre el frame que uno quiere con la sintaxis de “index x” donde x deber ser el número del frame que el programador desea mostrar en la interfaz. Entonces por ejemplo si tengo un archivo GIF el cual ya se encuentra animado y quiero mostrar el tercer frame en la interfaz a la hora de mostrar la imagen en tkinter uso el formato “index 3”. Además, se trató sin éxito de usar el método format para tratar de hacer el index una variable y poder cambiarla en cierto tiempo, pero no se tuvo éxito; en esto se duró 27 horas en un lapso de 2 días. Al tercer día decidí crear la animación yo, cambiando la imagen cada cierto tiempo, lo cual logré después de 10 horas de intentar poder hacer la recursividad con éxito. Para lograr la animación lo que hago es crear una lista con los nombres de las imágenes que equivalen a los frames de un GIF ya animados. Luego partiendo que ya existe un Label en la interfaz se define una función para cada animación que use dos variables globales, una que equivale a la posición de la lista con los nombres de los frames y otra de repeticiones por si quiero la animación se repita otra vez desde el inicio y ambos variables son iguales a -1. Luego uso dos condicionales if: el primero para limitar el número de repeticiones y así terminar la animación y el segundo para que cuando ya haya mostrado todos las frames vuelva a empezar y por último se utiliza el método after para adaptar el tiempo de cada animación.
3. Límite de recursión: Al probar todas a las animaciones se llegó al límite de animaciones, sin embargo, como se decidió usar recursividad y no iteración entonces tuve que importar sys y aumentar el límite.
4. Versión de tkinter y PIL: A la hora de exportar de Sketch mi Robot salen con formato PNG y yo tenía entendido que tkinter no acepta PNG. Entonces para tratar de no convertir todas las imágenes a GIF decidí usar PIL, sin embargo, no logré implementarlo bien entonces no logré que la animación usará fotos que abría con PIL. Entonces seguí investigando de cómo hacer para que tkinter acepte PNG y me di cuenta que mi computadora usa tcl 8.5 pero existe tcl 8.6 que si lo acepta entonces cambiando mi computadora de Mac OS a Windows logré usar tcl 8.6 y poder usar PNG, porque era muy inestable y complicado actualizar tcl en Mac OS.
5. Ventana Help: A la hora de crear una nueva ventana con información para el usuario se decidió utilizar el widget text para el cual se investigó y se invirtió su tiempo para darle formato por medio de tags al contenido. Además, se implementaron fuentes de Google de libre uso.
6. Vincular el texto del Entry con las animaciones: Después de leer de diferentes fuentes como utilizar el widget Entry se logró crear la función read(entry) la cual revisa el contenido del entry para accionar alguna función si concuerdan los nombres.
7. Vincular el Entry con el teclado: Al inicio se activaba read(entry) con un botón, pero luego se decidió tratar de vincularlo con la tecla enter, desafortunadamente se gastó tiempo en descubrir que enter y return son la misma tecla, también se intentó que a la hora de apretar enter y accionar read(entry) que los datos del entry se borraran, pero no se logró.
8. Animación con sonido: Se invirtió bastante tiempo en tratar de usar mp3 con Snack, PyAudio y Pyglet por las ventajas de utilizar mp3 en lugar de wav, principalmente calidad y poco espacio. Desafortunadamente no se logró, y se terminó utilizando winsound, entonces debido a esto en el archivo readme.txt se especifica que es necesario utilizar Windows. Luego se buscó en YouTube y algunos generadores de voz que no termine utilizando por su mala calidad archivos de audio. Después se extrajo el audio de los videos, y se convirtió en wav. Luego se probó repetidas veces las animaciones con los archivos de audio para lograr cortarlas y así “sincronizarlas” con la animación y para que así ocuparan menos espacio.
9. Energía: Este fue el reto más difícil que se encontró, primero se investigó para poder definir una variable global como Intvar() ya que es un número. Luego se buscaron diseños para la batería y sus diferentes estados. Más adelante se definió la función base del problema de energía que es battery\_level(), la cual revisa en cada animación el nivel de batería y cambia la imagen de la batería, además sirve como guía para dar algunas alertas. Luego sigue power(n), el cual luego de algunas horas se logró poder extraer solo los números de la entrada power(n) donde n es un numero con slicing, sin embargo, mi conocimiento previo de strings era poco entonces tuve que invertir algunas horas en eso para descubrir que se puede hacer slicing con strings y principalmente que se puede con negativos para eliminar el ultimo digito. Entonces luego de lograr power(n) tuve que modificar read(entry) para que power(n) funcionara en su totalidad, lo cual no fue rápido. Finalmente implementar la variable global energy en todas las animaciones que modifican la energía.
10. Guardar el estado: Luego de investigar sobre el modulo OS, Pickle y CSV se trató hasta el final del tiempo permitido, para poder guardar la variable energy, sin embargo, no se tuvo éxito.

# Análisis de Resultados:

El programa funciona y corre todas las animaciones. Además, acepta comandos en mayúscula y pide desde el inicio la batería para que el usuario decida cuanto quiere que el robot tenga desde un principio. Cuando llego el tiempo de pruebas al final tuvieron éxito, solo me quedo una duda y algo que no logre. Mi duda es la forma de hacer las animaciones ya que tuve que aumentar el límite de recursividad y lo que no logre fue guardar el estado del robot.

# Bitácora de actividades:

|  |  |  |
| --- | --- | --- |
| Tipo de Actividad | Descripción | Duración |
| Investigar | Investigar sobre vector graphics para diseñar el robot | 7 horas |
| Diseño | Crear el diseño del robot en Sketch | 6 horas |
| Diseño | Crear GIFs animados con las imágenes del robot | 2 horas |
| Investigar | Investigar sobre modulo os, PIL y como instalar librerías externas con pip | 6 horas |
| Investigar | Investigar sobre módulo pickle, csv, winsound, manipulación de archivos con Python | 7 horas |
| Programación | Empezar la base del interfaz | 2 horas |
| Investigar | Buscar sobre Pyglet, Snake, PyAudio | 4 horas |
| Investigar | Buscar como reproducir GIF ya animados | 27 horas |
| Investigar | Buscar como actualizar la versión de tkinter a 8.6 | 1 horas |
| Investigar | Investigar cómo utilizar text widget y Google Fonts | 1 horas |
| Programación | Desarrollar la ventana de ayuda | 5 horas |
| Análisis de Requerimientos | Se decidió ampliar la ventana principal, utilizar winsound, y utilizar PNG | 1 hora |
| Investigar | Como cambiar varias imágenes usando after | 5 horas |
| Programar | Desarrollar las animaciones sin energía o sonido en una ventana aparte sin necesidad de entry | 4 horas |
| Investigar | Buscar como vincular los entry | 1 hora |
| Programar | Crear la función read(entry) y vincularla con las animaciones | 2 horas |
| Investigar | Investigar como vincular con el teclado el entry | 2 horas |
| Programar | Vincular el entry con el teclado y empezar a crear Threads para la música | 2 horas |
| Diseño | Buscar archivos de audio en WAV para las animaciones | 2 horas |
| Programar | Implementar los audios y luego cortarlos | 2 horas |
| Diseño | Buscar diseños para la batería y sus estados | 1 hora |
| Programar | Crear Label de Batería, level\_battery() para que cambie los estados, power(n), status | 9 horas |
| Investigar | Slicing con Strings para power(n) en el entry | 1 hora |
| Programar | Terminar power(n) e implementar la energía en las animaciones | 3 horas |
| Análisis de Requerimientos | Comprimir las imágenes y audio | 1 hora |
| Investigar | Investigar aún más sobre OS, Pickle para guardar el estado | 5 horas |
| Documentación Interna | Documentar el código fuente | 3 horas |
| Pruebas | Probar el código fuente con diferentes secuencias de comandos y niveles de energía | 4 horas |
| Programar | Corregir cambios de la energía por las animaciones left() y right() que descubrí con las pruebas | 1 hora |
| Elaboración del documento | Redactar este documento | 2 horas |
| Programar | Seguir intentando sin éxito guardar el estado de la energía | 2 horas |

# Estadística de tiempos:

|  |  |
| --- | --- |
| **FUNCION** | **Tiempo** |
| Análisis de requerimientos | 2 horas |
| Diseño de la aplicación | 11 horas |
| Investigación de funciones | 67 horas |
| Programación | 32 horas |
| Documentación Interna | 3 horas |
| Pruebas | 4 horas |
| Elaboración del documento | 2 horas |
| **TOTAL** | 121 horas |

# Conclusión Personal:

Mi conclusión es que este proyecto programado estuvo muy bien ideado ya que aprendí mucho y siento que al final cree algo interesante y divertido. Sin embargo, siento que a veces es muy ambiguo y por eso invertí demasiado tiempo investigando cosas que no me sirven y me atrasaron con el progreso del proyecto en sí.