INDICE

PORTADA

Programación Básica

Proyecto DREAM WORLD CASINO

Profesor:
Alvaro Camacho

Integrantes:

Kevin Chinchilla
Jafet Rivera
Edgar Bonilla

22/8/2023

Objetivo general

El objetivo principal y ambicioso de este código radica en fusionar la educación y el entretenimiento al crear una simulación de casino. Esta simulación aspira a servir como una plataforma de aprendizaje interactiva para aquellos que se inician en la programación, brindándoles la oportunidad de explorar y dominar los conceptos fundamentales de codificación mientras participan en juegos de casino cautivadores como el Blackjack y las tragamonedas. De esta manera, se busca que los principiantes no solo adquieran habilidades técnicas valiosas, como el manejo de datos, el diseño de interfaces y la estructuración de lógica, sino que también experimenten una experiencia envolvente que combina la adquisición de conocimientos con la diversión y el entretenimiento

implementados

Objetivos Específicos

Objetivo Específico 1: Fomentar la comprensión de los conceptos de programación a través de la interacción con juegos de casino.

Diseñar el juego del Blackjack de manera que los jugadores deban implementar lógica de control de flujo para tomar decisiones sobre pedir cartas o plantarse.

Implementar el juego de tragamonedas para que los principiantes practiquen el uso de variables, generación aleatoria y estructuras de repetición.

Proporcionar comentarios detallados en el código para explicar cómo funcionan las diferentes partes y cómo se aplican los conceptos de programación en el contexto de los juegos.

Objetivo Específico 2: Facilitar la exploración práctica de conceptos de interfaz de usuario y manipulación de datos.

Configurar una interfaz de usuario amigable que permita a los usuarios realizar acciones como apostar, retirar dinero y tomar decisiones de juego.

En el juego de Blackjack, mostrar las cartas del jugador y del crupier de manera ordenada y comprensible en la interfaz.

En la tragamonedas, presentar los resultados de cada ronda de juego en la interfaz de manera visualmente atractiva.

Ofrecer opciones para que los usuarios seleccionen monedas y cantidades de apuesta en el juego de tragamonedas, demostrando la manipulación de datos en tiempo real.

Indicadores de Logro:

Los usuarios pueden describir cómo se creó la interfaz de usuario y cómo se integran las funciones de juego en ella.

Los jugadores pueden interactuar exitosamente con la interfaz para realizar acciones como realizar apuestas, visualizar resultados y tomar decisiones de juego.

Requerimientos del Sistema

Interfaz de Usuario Amigable:

La simulación debe presentar una interfaz gráfica que permita a los usuarios acceder a las diferentes opciones y funciones de manera intuitiva. Los elementos de la interfaz deben estar organizados de manera coherente y ser visualmente atractivos.

Una interfaz amigable y visualmente atractiva es esencial para asegurar que los usuarios, especialmente aquellos nuevos en programación, puedan navegar sin dificultades por las diferentes funciones y opciones disponibles.

Registro de Usuarios:

El sistema debe permitir a los usuarios registrarse proporcionando un nombre de usuario único y un PIN seguro. Además, el sistema debe verificar la validez del ID y garantizar la privacidad y seguridad del PIN durante el proceso de registro.

El registro de usuarios es un paso crucial para la personalización de la experiencia de juego. Garantizar la unicidad del ID y la seguridad del PIN es esencial para mantener la integridad de las cuentas de los usuarios.

Juegos de Casino Implementados:

La simulación debe incluir dos juegos de casino completamente funcionales: Blackjack y tragamonedas. Cada juego debe seguir las reglas y la lógica tradicionales asociadas con su respectiva modalidad de juego.

La implementación precisa de los juegos es esencial para ofrecer una experiencia de casino auténtica y permitir que los usuarios participen activamente en actividades de juego realistas.

Lógica del Juego:

Los juegos de casino (Blackjack y tragamonedas) deben ser programados de acuerdo con las reglas y la lógica intrínseca de cada juego. Esto incluye aspectos como el cálculo de resultados, la determinación de victorias o pérdidas, y la asignación de premios.

La fidelidad en la lógica de juego es fundamental para asegurar que los usuarios experimenten situaciones de juego auténticas. Además, brinda a los principiantes en programación la oportunidad de aplicar lógica condicional y matemática en un contexto práctico.

Manipulación de Datos y Variables:

El sistema debe gestionar y manipular datos como saldos de usuarios, apuestas y resultados de juegos. Debe utilizar variables para rastrear y actualizar estas cantidades a medida que los usuarios interactúan con los juegos.

La manipulación efectiva de datos y variables es fundamental para llevar a cabo transacciones de dinero, ajustar saldos y registrar resultados. Esta funcionalidad permite a los principiantes en programación comprender cómo se gestionan los datos en aplicaciones reales.

Manejo de Archivos:

El sistema debe ser capaz de crear, leer y escribir archivos que almacenan información relevante de usuarios, como nombres y saldos. Estos archivos deben mantenerse actualizados a medida que los usuarios interactúan con el sistema.

El manejo de archivos es esencial para garantizar la persistencia de los datos de usuarios entre sesiones de juego. Esto asegura que los usuarios puedan retomar sus actividades de juego desde donde las dejaron.

Feedback y Comunicación con el Usuario:

El sistema debe proporcionar retroalimentación continua a los usuarios, incluyendo la visualización de saldos actuales, resultados de juegos y mensajes de estado. La interfaz debe ser interactiva y responder en tiempo real a las acciones del usuario.

La retroalimentación es clave para informar a los usuarios sobre el estado de sus actividades y decisiones. Una comunicación efectiva promueve una experiencia fluida y comprensible.

Control de Flujo y Estructuras de Bucles:

El código debe emplear estructuras de control de flujo como condicionales y bucles para guiar las acciones del usuario y determinar el flujo de juego. Estas estructuras deben permitir a los usuarios tomar decisiones y avanzar en el juego de manera coherente.

El uso adecuado de control de flujo y bucles es fundamental para dirigir el comportamiento del sistema en respuesta a las interacciones del usuario. Proporciona una experiencia interactiva y adaptativa.

En conjunto, estos Requerimientos del Sistema constituyen una base sólida para la simulación de casino, que tiene como objetivo educar a los principiantes en programación y entretener a los usuarios al brindarles una experiencia de casino virtual auténtica. Cada uno de estos requisitos contribuye a la creación de una experiencia completa y funcional que permite a los usuarios aprender y aplicar conceptos de programación mientras disfrutan de juegos de casino populares.

Problema

El código proporcionado aborda varias problemáticas y desafíos, y aporta soluciones específicas que tienen un enfoque educativo y de entretenimiento. Algunas de las problemáticas que resuelve incluyen:

Aprendizaje de Programación: Para los principiantes en programación, es esencial contar con ejemplos prácticos y aplicables que les permitan comprender los conceptos teóricos en un contexto real. El código proporcionado crea una simulación interactiva que utiliza juegos de casino populares para enseñar y reforzar conceptos como variables, lógica condicional, bucles y manipulación de datos. A través de la interacción con la simulación, los usuarios pueden aplicar lo que han aprendido de manera efectiva, lo que mejora su comprensión y habilidades en programación.

Mejora de la Lógica y Estructuras de Software: Uno de los mayores desafíos para los nuevos programadores es desarrollar la habilidad de diseñar y crear software con una estructura lógica y eficiente. El código brindado presenta juegos de casino que requieren una lógica cuidadosa para determinar los resultados de las acciones de los usuarios. Al interactuar con el código y sus juegos, los usuarios se enfrentan a la necesidad de pensar de manera analítica y estructurada para implementar las reglas y la lógica del juego de manera coherente.

Comprensión de Interacción de Usuario: Diseñar sistemas que respondan a la interacción del usuario es fundamental en el desarrollo de software interactivo. El código proporcionado muestra cómo crear una interfaz de usuario amigable que permita a los usuarios interactuar con diferentes opciones y funciones. A través de la simulación, los usuarios pueden experimentar la importancia de la comunicación y retroalimentación en tiempo real, lo que mejora su comprensión de la relación entre el usuario y el software.Práctica de Manipulación de Datos: La manipulación de datos es una habilidad fundamental en programación. El código muestra cómo gestionar y almacenar datos de usuarios, saldos y resultados de juegos utilizando variables y archivos.

Los usuarios pueden practicar la actualización y el seguimiento de datos, lo que les brinda una valiosa experiencia en la gestión de información en aplicaciones reales motivación y entretenimiento: Aprender programación puede ser desafiante, pero también puede ser divertido. El código brindado integra juegos de casino emocionantes, como Blackjack y tragamonedas, para mantener a los usuarios comprometidos y entretenidos mientras aplican conceptos de programación en un entorno práctico. Esto ayuda a mantener la motivación y el interés de los usuarios en su aprendizaje.

Estructuras de software requeridas y algoritmos

Función main():
La función main() sirve como punto de entrada del programa. Desde aquí se controla la ejecución de diversas funcionalidades y se coordina la interacción con el usuario en el casino virtual.
Clase DreamWorld Casino:

La clase Dream World Casino es el corazón del programa, representando el entorno completo del casino virtual. Contiene una serie de métodos que facilitan la administración de usuarios, depósitos, juegos y más.
Método __init__(self):

Este método es invocado cuando se crea una instancia de DreamWorld Casino. Establece valores iniciales, como usuarios registrados y montos de depósito.
Método iniciar(self):

El método iniciar() muestra un menú principal al usuario. A través de este menú, los usuarios pueden acceder a diferentes partes del casino, como el registro, el casino en sí y la configuración.
Método guardar InfoUsuario(self, idUsuario):

Este método se encarga de crear carpetas individuales para cada usuario, donde se almacena información específica de ellos. Esto se lleva a cabo durante el proceso de registro.
Método eliminar Usuario(self):

eliminar Usuario() permite a los usuarios eliminar sus cuentas, siempre que no tengan saldo pendiente. La función solicita credenciales y realiza la eliminación si se cumplen las condiciones.
Método obtener Monto Mínimo(self):

obtenerMontoMinino() retorna el monto mínimo requerido para realizar un depósito en el casino. Esto asegura que los usuarios cumplan con un monto mínimo antes de jugar.
Método submenu Casino(self):

Este método presenta un submenú específico del casino. Ofrece opciones como retirar dinero, depositar, jugar juegos, eliminar usuarios y más.
Método jugar Blackjack(self):

jugar Blackjack() implementa el juego de Blackjack. Los jugadores pueden hacer apuestas, recibir cartas, tomar decisiones estratégicas y enfrentarse al crupier.
Método jugar Tragamonedas(self):

jugar Tragamonedas() simula el juego de tragamonedas en el casino. Los usuarios apuestan, jala una palanca y pueden ganar premios según los símbolos obtenidos.
Método menú Traga(self):

menú Traga() presenta el submenú específico para el juego de tragamonedas. Los usuarios pueden acceder a instrucciones y jugar en una simulación de máquina tragamonedas.
Método depósito Obligatorio(self):

depósito Obligatorio() guía a los usuarios a través del proceso de realizar un depósito obligatorio antes de comenzar a jugar. Asegura que los usuarios tengan fondos suficientes para participar.
Método salir(self):

salir() muestra un mensaje de despedida cuando los usuarios deciden salir del casino virtual. Finaliza la ejecución del programa de manera controlada.
Método registroUsuario Nuevo(self):

registroUsuario Nuevo() permite a los usuarios registrarse en el casino virtual. Los usuarios eligen un nombre de usuario y un PIN, y se almacenan en la lista de usuarios registrados.
Método guardar InfoUsuario(self, idUsuario):

Descripción: guardar InfoUsuario() crea carpetas individuales para cada usuario, donde se almacena información específica de ellos. Esto forma parte del proceso de registro.
Método menu_principal():

menu_principal() presenta el menú principal del programa. Los usuarios pueden seleccionar entre opciones como registro, acceso al casino y configuración.
Funciones de Manipulación de Archivos agregarNombre() y mostrarInformacion():

Estas funciones permiten a los usuarios agregar un nombre de usuario y mostrar la información almacenada en un archivo de texto, brindando una interacción sencilla pero didáctica.Dawson, M. (2011).
Funciones crearMano() y mezclarNaipe(naipe):

Estas funciones contribuyen a la lógica del juego de Blackjack. crearMano() crea una mano de cartas, mientras que mezclarNaipe(naipe) las baraja aleatoriamente.Dawson, M. (2011).
Funciones valorDeMano(mano) y mostrarMano(mano, ...):

Estas funciones son esenciales para el juego de Blackjack. valorDeMano(mano) calcula el valor de una mano de cartas, y mostrarMano(mano, ...) muestra las cartas en pantalla de manera legible para el jugador.
Cada una de estas partes, desde el menú principal hasta los algoritmos específicos para juegos y funciones de manipulación de archivos, contribuye en conjunto a una experiencia de usuario interactiva y educativa, permitiendo a los usuarios aprender programación mientras se divierten en el casino virtual

Bibliotecas/Estructuras complementarias

import random:
La biblioteca random proporciona funciones para generar números pseudoaleatorios. Se utiliza para simular la distribución aleatoria de cartas en los juegos de casino, como Blackjack y tragamonedas.Zelle, J. M. (2010)
import time:
La biblioteca time proporciona funciones para trabajar con el tiempo. Se utiliza para introducir pausas en el flujo del programa, creando efectos de espera realistas, como cuando se muestran los resultados de los juegos.
import os:
La biblioteca nos proporciona funciones para interactuar con el sistema operativo. En este código, se utiliza para crear carpetas individuales para cada usuario registrado, donde se almacena información específica.
import getpass:
La biblioteca getpass proporciona una función para ocultar la entrada de texto en la consola, como al ingresar contraseñas o pines. Se utiliza para solicitar información confidencial, como pines de usuario.Zelle, J. M. (2010)
Todas la bibliotecas utilizadas fueron implementadas en clase y estaban en el documento de la guía del lenguaje de programación de PYTHON.

Implementación de Blackjack en Python

Este informe describe la implementación de un juego de Blackjack en Python, siguiendo una estructura modular y utilizando conceptos de programación estructurada. Se presenta una solución de código detallada para el juego, que permite a los jugadores competir contra un crupier en una simulación interactiva del popular juego de cartas.

Solución de Problemas

Durante el proceso de implementación del juego de Blackjack, se enfrentaron varios desafíos que requirieron soluciones creativas. Uno de los desafíos clave fue la necesidad de mantener la segunda carta del crupier oculta hasta el final del juego. Para abordar este problema, se introdujo un parámetro adicional en la función mostrarMano() llamado revelarSegundaCarta. Esto permitió mostrar y ocultar la segunda carta del crupier según avanza el juego.

Explicación del Código

El código se estructura en torno a funciones definidas, lo que facilita la comprensión y modificación del juego.

Creación y Mezcla del Naipe: La función crearNaipe() combina los diferentes naipes y valores para crear un naipe completo. La función mezclarNaipe(naipe) reorganiza el naipe de manera aleatoria, garantizando una distribución justa de las cartas.

1. Cálculo del Valor de la Mano: La función valorDeMano(mano) calcula el valor total de las cartas en una mano, teniendo en cuenta la posibilidad de que el As tenga un valor de 11 o 1. Esto asegura que se gestione correctamente la puntuación de las manos de los jugadores y del crupier.
2. Visualización de la Mano: La función mostrarMano(mano, ocultarPrimeraCarta=False, jugador=True, revelarSegundaCarta=False) se encarga de mostrar las cartas de un jugador o el crupier. La capacidad de ocultar la segunda carta del crupier y revelar solo al final del juego.
3. Turno del Jugador y Crupier: Los turnos del jugador y el crupier se manejan con bucles y decisiones basadas en la entrada del jugador. Las cartas se entregan y se muestran conforme a las acciones del jugador, y el crupier toma decisiones estratégicas según las reglas del Blackjack y también se basa en el import random que genera de manera aleatoria el juego.
4. Determinación del Resultado y Actualización del Saldo: La función jugarBlackjack() controla la lógica general del juego. Compara los valores de las manos del jugador y el crupier para determinar el resultado de la partida y actualiza el saldo del jugador en consecuencia.

Juego de Tragamonedas - Algoritmo y Solución de Problemas

Algoritmo

1. Importar Módulos: Importar los módulos time y random para gestionar el tiempo y generar números aleatorios respectivamente.

2. Función mostrar_instrucciones(): Definir una función para mostrar las instrucciones del juego. Esto proporciona una guía al jugador sobre cómo jugar el juego.
3. Función jugar_tragamonedas(saldo): Función que permite jugar tragamonedas. La función toma el saldo actual del jugador como entrada.
a. Verificar Saldo: Verificar si el saldo es suficiente para jugar ($100 o más). Si no, mostrar un mensaje de advertencia y devolver el saldo sin cambios.
b. Iniciar el Juego: Mostrar un mensaje de bienvenida al jugador e indicar el saldo actual.
c. Bucle Principal: Iniciar un bucle que permite al jugador jugar varias rondas mientras decidan continuar.
i. Input de Juego: Solicitar al jugador que presione la tecla "J" para jalar la palanca y comenzar la ronda.
ii. Generar Símbolos: Generar tres símbolos aleatorios para representar la ronda actual. Los símbolos posibles son '@', '#', '+', '7'.
iii. Evaluar Resultados: Contar la cantidad de símbolos iguales en la ronda actual y ajustar el saldo en función de los resultados.
iv. Verificar Saldo: Verificar si el saldo restante es suficiente para continuar jugando. Si no, mostrar un mensaje de saldo insuficiente y salir del bucle.
v. Decisión de Continuar: Preguntar al jugador si desean seguir jugando. Si la respuesta no es "S" o "s", salir del bucle de juego.
4. Función menu_Traga(): Definir una función para mostrar el menú principal y controlar el flujo del juego.
a. Saldo Inicial: Establecer el saldo inicial del jugador en $1000.
b. Bucle del Menú: Iniciar un bucle que muestra las opciones del menú principal y permite al jugador elegir una acción.
i. Mostrar Opciones: Mostrar las opciones disponibles en el menú principal: "Mostrar instrucciones", "Jugar a la tragamonedas" y "Salir".
ii. Input de Selección: Solicitar al jugador que seleccione una opción ingresando el número correspondiente.
iii. Ejecutar Opción: Ejecutar la acción correspondiente según la selección del jugador: mostrar instrucciones, jugar a la tragamonedas o salir del juego.
5. Iniciar Juego: Llamar a la función menu_Traga() para comenzar el juego y permitir que los jugadores interactúen con el menú principal.

Solución de Problemas - Explicación de Código

1. Importar Módulos (import time y import random): Se importan los módulos necesarios para el juego. time se utiliza para gestionar el tiempo de espera y random para generar números aleatorios para los símbolos.
2. Función mostrar_instrucciones(): Define una función para mostrar las reglas del juego. Esto asegura que los jugadores tengan acceso a las instrucciones antes de comenzar.

3. Función jugar_tragamonedas(saldo): Esta función permite a los jugadores jugar al juego de tragamonedas. Se utiliza para gestionar la lógica del juego y actualizar el saldo del jugador según los resultados.
4. Función menu_Traga(): Define una función que actúa como el menú principal del juego. Permite a los jugadores elegir entre diferentes opciones, como jugar o salir del juego.
5. Estructura de Control (if, while, for): Se utilizan estructuras de control para tomar decisiones y ejecutar acciones en función de ciertas condiciones. Esto permite que el juego responda de manera adecuada a las acciones del jugador.
6. Interacción con el Jugador (input, print): Se utiliza la función input() para permitir al jugador interactuar con el juego. Los mensajes print() se usan con el fin de proporcionar información al jugador.
7. Generación Aleatoria (random.choice()): La función random.choice() se utiliza para seleccionar cartas aleatorias a como avanza el juego.
8. Espera de Tiempo (time.sleep()): La función time.sleep() se usa para crear una pausa en el juego después de mostrar cada símbolo en la pantalla. Esto para poder plantar como si estuviera viendo la ruleta girar en un casino de verdad
9. Actualización de Saldo: El saldo del jugador se actualiza en función de los resultados de cada ronda. Si el jugador gana, su saldo aumenta; si pierde, disminuye.

Manual de Usuario: DreamWorld Casino

Este manual es para utilizar el Dreamworld Casino ya que es una programa sencillo y para principiantes que deseen aprender a jugar diferentes programas de un casino

1. Introducción:

2. Registro y Depósito:

Antes de comenzar a jugar, necesitas registrarte y hacer un depósito en tu cuenta. Sigue estos pasos:

Paso 1: En el menú principal, selecciona "Registro de Usuario Nuevo".

Paso 2: Ingresa un nombre de usuario (ID) y un PIN de seguridad. Asegúrate de recordarlos para futuros accesos.
Paso 3: Una vez registrado, selecciona "DreamWorld Casino" en el menú principal.
Paso 4: Elige "Depósito Obligatorio" para agregar fondos a tu cuenta. Sigue las instrucciones y selecciona la moneda en la que deseas hacer el depósito.
3. Explorando el Casino:

Una vez registrado y con fondos en tu cuenta, selecciona "DreamWorld Casino" en el menú principal para explorar el casino virtual.
4. Juegos:

El casino ofrece emocionantes juegos que puedes disfrutar:

4.1. Blackjack:

Selecciona "Jugar al Blackjack" en el menú del casino.
Sigue las instrucciones en pantalla para hacer apuestas y jugar contra el crupier.
Toma decisiones estratégicas y disfruta del emocionante juego de cartas.
4.2. Tragamonedas:

Selecciona "Jugar a la Tragamonedas" en el menú del casino.
Aprende las instrucciones del juego y realiza apuestas.
Jala la palanca virtual para revelar símbolos y descubre si ganas premios.
5. Cierre y Despedida:

Cuando decidas terminar tu sesión en DreamWorld Casino, puedes seleccionar "Salir" en el menú principal. Nos encantó tenerte aquí y esperamos que hayas tenido una experiencia emocionante.

Instrucciones del tragamonedas:
Debes apostar al menos $100 para jugar.
Presiona la tecla J para jalar la palanca e iniciar el juego.
Cada ronda se mostrarán tres símbolos en pantalla.
Si aparecen tres @ en una ronda, recuperarás tu apuesta.
Si aparecen tres # en una ronda, ganarás el doble de tu apuesta.
Si aparecen tres 7 en una ronda, ganarás el acumulado.
El juego continuará hasta que decidas dejar de jugar.

Instrucciones del Black Jack:
Iniciar Juego:
Selecciona "Jugar al Blackjack".
Hacer Apuesta:

Ingresa tu apuesta (cantidad de dinero).
Recibir Cartas:
Recibirás 2 cartas visibles.
Tomar Decisiones:
Elige "Pedir" para más cartas o "Parar" para quedarte.
Ganar o Perder:
Ganas si tus cartas suman menos de 21 y son mejores que las del crupier.
No te pases de 21.
Resultado Final:
Si ganas, ganas dinero; si pierdes, pierdes tu apuesta.
Decide si juegas otra ronda o sale
Conclusiones

Se halogrado adquirir una amplia gama de habilidades y conocimientos que son fundamentales en el mundo de la programación. Este proyecto nos ha permitido no solo explorar las capacidades de Python, sino también comprender cómo aplicar esos conocimientos en la creación de un casino simulado con múltiples funcionalidades.

En el transcurso de este proyecto, hemos experimentado con conceptos esenciales de programación que incluyen la manipulación de estructuras de datos, como listas y diccionarios, así como el control de

Para dar vida a nuestro casino simulado, hemos aprovechado algunas de las bibliotecas proporcionadas por Python. Utilizamos la biblioteca random para generar eventos aleatorios en nuestros juegos, la biblioteca time para controlar la velocidad de visualización en el juego de tragamonedas, y la biblioteca os para gestionar la creación de carpetas y archivos para almacenar información de los usuarios.

El propósito central de este código es brindar una experiencia interactiva en la que los usuarios puedan registrarse, sumergirse en la emoción de los juegos de casino como el blackjack y la tragamonedas, administrar su saldo de manera realista, y explorar una variedad de características del casino. Más allá de ser solo un ejercicio de programación, este proyecto también nos ha permitido aplicar la lógica de los juegos y crear una interfaz de usuario simple pero atractiva.

En resumen, este proyecto ha sido un paso significativo en nuestro viaje de aprendizaje en programación. Hemos ganado confianza en la creación y estructuración de programas, así como en la implementación de características interactivas y funcionales. Al completar este proyecto, hemos establecido una base sólida sobre la cual podemos construir proyectos más complejos en el futuro. Estamos emocionados por continuar explorando nuevas áreas y desafíos en el mundo de la programación.

Bibliografía

Referencias

1. Python. (s.f.). Documentación oficial de Python. https://docs.python.org/
2. Random. (s.f.). Módulo random en Python. https://docs.python.org/3/library/random.html
3. Time. (s.f.). Módulo time en Python. https://docs.python.org/3/library/time.html
4. OS. (s.f.). Módulo os en Python. https://docs.python.org/3/library/os.html
5. Getpass. (s.f.). Módulo getpass en Python. https://docs.python.org/3/library/getpass.html
6. How to Think Like a Computer Scientist - Interactive Edition. (s.f.). http://interactivepython.org/
7. Dawson, M. (2011). Python Programming for the Absolute Beginner (3rd ed.). Cengage Learning.
8. Zelle, J. M. (2010). Python Programming: An Introduction to Computer Science (2nd ed.). Franklin, Beedle & Associates.