
Consideraciones Generales

- El proyecto debe ser entregado en grupos de máximo 2 personas (se puede trabajar de forma individual).
- Cada grupo deberá tener un nombre propio.
- Cada grupo deberá darle un nombre a su Aplicación.
- Se debe entregar el código correspondiente a la actividad en la plataforma BrightSpace, tanto en PDF (en un archivo diferente), como en módulo de Python (.py).
- Se debe sustentar el proyecto (última semana de parciales).
- Cualquier caso de fraude se penalizará con 0.0 y apertura de proceso disciplinario.

COMPONENTE OBLIGATORIO

Sección I. Marco Teórico y Planeación:

1. (0.5 / 5.0) – ¿Qué es BLAST y para qué es usado?, describa la aplicación web “Standard Nucleotide BLAST.
2. (0.5 / 5.0) – Construya un diagrama de flujo del procedimiento que va a seguir para implementar la aplicación en la sección II.

Sección II. Implementación de aplicación

1. (1.5 / 5.0) – Menú Principal.

   El sistema debe incluir un menú principal, el cual deberá mostrar al usuario las opciones que tiene disponible para conocer más acerca del virus COVID-19. El menú principal deberá mostrarse de la siguiente forma:

   ----- Nombre_grupo -----
   ----- Nombre_app ------
   ----- Menú Principal -----
   1. ¿Qué es COVID-19?
   2. ¿Qué síntomas causa el COVID-19?
   3. ¿Qué es el SARS-CoV-2?
   4. Secuencia genética del SARS-CoV-2.
   5. Análisis de Secuencias Genéticas.
   6. Cerrar programa.

El menú principal deberá estar contenido en la función llamada “mainMenu()”.

Después de mostrar el menú, el programa deberá solicitar ingresar el número de una opción al usuario, si el usuario selecciona una de las opciones de 1 a 4, el sistema deberá mostrar la información correspondiente en pantalla y deberá solicitar al usuario “Ingrese cualquier valor para volver al menú principal”.

Si el usuario selecciona la opción 5, el programa deberá mostrar el menú de análisis de secuencia genética (ver Sección II, punto 2).

Si el usuario selecciona la opción 6, el programa deberá imprimir en pantalla “Fin del Programa” y seguidamente deberá terminar.

Si el usuario selecciona una opción diferente a las mencionadas, el sistema deberá mostrar en pantalla “ERROR: Selección Inválida” y deberá volver a mostrar el menú principal.

2. (2.5 / 5.0) – Menú de Secuencia Genética.

Cuando se seleccione la opción 5 en el menú principal, el programa deberá mostrar en pantalla el siguiente Menú:

----- Nombre_grupo -----
----- Nombre_app ------
----- Menú Secuencia Genética -----
1. Crear Secuencia Genética.
2. Eliminar Secuencia Genética.
3. Buscar Secuencia Genética.
4. Comparar Secuencia Genética.
5. Construir Secuencia Genética Complementaria.
6. Mostrar secuencias genéticas almacenadas.
7. Regresar al menú anterior.

Seguidamente, el programa deberá solicitarle al usuario que elija una opción.

El menú de secuencia genética deberá ser una función que reciba un parámetro llamado “genetic_list” que corresponde a una lista donde se guardarán las secuencias genéticas.

Requerimientos Menú Secuencia Genética:

- Prototipo: geneticSequenceMenu(genetic_list).
- La función recibe un solo parámetro que corresponde a una lista de secuencias genéticas. Este argumento siempre será una lista que

siempre va a contener cadenas de texto compuestas únicamente combinaciones de los caracteres “A”, “T”, “C”, “G” o “A”, “U”, “C”, “G”. Esta lista puede estar vacía.

- Las opciones 1 a 6 corresponden a diferentes funciones como se explican a continuación:

  - Opción 1: Función Crear Secuencia Genética:
    - Prototipo: createSequence(genetic_list).

    - La función recibe un argumento “genetic_list” que tiene las mismas características del argumento de la función geneticSequenceMenu.

    - La función deberá solicitar al usuario: “¿Crear secuencia de forma automática? [Y/N]”. El usuario solamente deberá ingresar una de las dos opciones.

    - Si el usuario introduce “Y” o “y”, la función deberá crear una secuencia de ADN o ARN de forma aleatoria. El programa deberá preguntar por la longitud de la secuencia (valor mínimo: 20).

      - La función deberá solicitar al usuario: “¿Crear secuencia de ADN o ARN?”. El usuario deberá asegurarse de ingresar una de las dos opciones.

    - Si el usuario introduce “N” o “n”, la función deberá solicitar al usuario la secuencia de ADN o ARN como corresponda.

    - El usuario deberá asegurarse de que la secuencia ingresada sea correcta.

    - Una vez que la secuencia genética esté completa, la función deberá anexarla al final de la lista “genetic_list”.

    - La función deberá mostrar en pantalla “Secuencia Agregada con Éxito:” y la secuencia agregada.

    - Por último, la función deberá retornar la lista “genetic_list”.

  - Opción 2: Función Eliminar secuencia.
    - Prototipo: deleteSequence(genetic_list).

- La función recibe un argumento “genetic_list” que tiene las mismas características del argumento de la función geneticSequenceMenu.
- Si la lista está vacía, la función deberá mostrar en pantalla: “ERROR: Lista Vacia” y deberá retornar genetic_list.
- La función deberá eliminar el último elemento de la lista.
- La función deberá mostrar en pantalla: “Última Secuencia Eliminada con Éxito”.
- La función deberá retornar genetic_list.

- Opción 3: Función Buscar una secuencia.
  - Prototipo: searchSequence(genetic_list, pattern).
  - La función recibe dos argumentos, el primero “genetic_list” tiene las mismas características del argumento de la función geneticSequenceMenu. El segundo argumento “pattern” es una cadena de texto que corresponde a una secuencia genética y está compuesto por una combinación de los caracteres “A”, “T” o “U”, “C” y “G”. Este argumento deberá ser solicitado al usuario cuando se seleccione la opción 3 en el menú de análisis de secuencias, el usuario deberá asegurarse de que este argumento sea correcto.
  - La función deberá recorrer la lista “genetic_list” y por cada elemento que sea exactamente igual al patrón, deberá mostrar su posición en pantalla de la forma: “Secuencia encontrada en la posición: ” [index].
  - Si el patrón no se encuentra en la lista genetic_list, la función deberá mostrar en pantalla: “No se encuentran resultados”.
  - Esta función no tiene retorno.

- Opción 4: Función comparar secuencia con parecido.
  - Prototipo: compareSequence(genetic_list, pattern, percent).
  - La función recibe 3 argumentos; el primero “genetic_list” tiene las mismas características del argumento de la función geneticSequenceMenu.
  - El segundo argumento “pattern” es una cadena de texto que corresponde a una secuencia genética y está compuesto por una combinación de los caracteres “A”, “T” o “U”, “C” y “G”.

Este argumento deberá ser solicitado al usuario cuando se seleccione la opción 4 en el menú de análisis de secuencias, el usuario deberá asegurarse de que este argumento sea correcto.

- El tercer argumento “percent” corresponde al porcentaje de parecido que desea buscar el usuario, este es un valor de tipo float que el usuario debe ingresar después de seleccionar la opción 4 en el menú de análisis de secuencias; el usuario debe asegurarse de que este valor se encuentre dentro del rango (0.0, 1.0).
- La función deberá recorrer toda la lista genetic_list y deberá mostrar en pantalla la posición de cada una de las secuencias contenidas que tengan un parecido mayor o igual al porcentaje ingresado por el usuario.
- Si la secuencia genética a revisar tiene una longitud diferente a la secuencia del patrón, su parecido será de 0%.
- La función deberá mostrar en pantalla: “Secuencia Parecida encontrada en: ” [index]
- Si la función no encuentra ningún elemento que tenga parecido con el patrón, deberá imprimir en pantalla: “No se encuentran resultados”.
- Esta función no tiene retorno.

- Opción 5: Función invertir cadena de ARN o complementar cadena de ADN.
  - Prototipo: def invertOrComplement(genetic_list).
  - La función recibe un argumento: “genetic_list”, el cual tiene las mismas características del argumento de la función geneticSequenceMenu.
  - Si la lista genetic_list está vacía, la función deberá imprimir en pantalla: “Lista Vacía”, seguidamente, deberá retornar genetic_list.
  - La función deberá tomar la última secuencia de genetic_list y deberá crear una cadena inversa si es de ARN o crear una cadena complementaria si es de ADN.

- La función deberá anexar al final de la lista genetic_list la cadena creada.
- La función deberá mostrar en pantalla: “Creación de Cadena Inversa Completa” si la cadena es de ARN o “Creación de Cadena Complementaria Completa” si la cadena es de ADN seguido de la cadena que fue creada.
- La función deberá retornar la lista genetic_list.

- Opción 6: Imprimir lista de secuencias.
  - La función geneticSequenceMenu deberá imprimir en pantalla todos los elementos de la lista genetic_list, elemento por elemento.
  - Si la lista está vacía, el programa deberá imprimir “Lista Vacía”.
- Opción 7: Cuando el usuario seleccione esta opción, la función geneticSequenceMenu deberá retornar la lista genetic_list.

Informática – Proyecto Final

OPTIONAL COMPONENT

This component is optional you can choose to complete it or not and it will not affect your final score negatively. If you choose to complete it, it will add to the score of the mandatory component up to 2.0.

1. (0.5 / 2.0). Translate everything on the mandatory component to English.
2. (1 .5 / 2.0) Modify some aspects of the mandatory component as follows:

   A. Option 1 Create sequence:
      - Make it so the function automatically verifies the user's input when the system asks about making the sequence automatically.
      - If the user inputs anything other than "Y", "y", "N" or "n"; the function must print on screen "Invalid Input" and ask again.
      - If the user inputs anything other than "DNA" or "RNA", the function must print on screen "Invalid Input" and ask again.
      - If the user chooses to input the genetic sequence themselves, the function must verify that the sequence is a valid one.

   B. Option 3. Search for a sequence:
      - The function must verify that the genetic sequence given by the user is a valid one.
      - If the sequence is not valid, the program must print on screen: "Invalid Sequence" and it must ask again for it.

   C. Option 4. Compare Sequences
      - The function must verify that the genetic sequence given by the user is a valid one.
      - If the sequence is not valid, the program must print on screen: "Invalid Sequence" and it must ask again for it.
      - The function must verify that the percentage given by the user is in the range (0.0, 1.0). If it's not, the system must print on screen: "Invalid Percentage" and it must ask for it again.