Estructura de Datos
Entrega 1, proyecto 2025-30
Esteban Blanco Madrid
Sophie Mejía Cuitiva

DOCUMENTO DE DISEÑO

Acta de Evaluación

- Se modificaron las funcionalidades de encontrar subsecuencia y enmascarar subsecuencias, de forma tal que encuentre y trabaje con las cantidades exactas existentes en un genoma, a diferencia de las versiones anteriores, ahora se aseguran de encontrar las cantidades exactas.

TADs

TAD Secuencia

Conjunto mínimo de datos:

- nombreS, cadena de caracteres, identifica la secuencia con un atributo que en este caso es su nombre
- bases, lista de caracteres, contiene todas las bases de una secuencia separando por líneas
- anchoBase, entero, representa el ancho de las líneas de la secuencia
- completa, booleano, indica si una secuencia está completa o no lo está si la secuencia tiene "-" o no

Operaciones:

- Secuencia(), crea una secuencia vacía
- Secuencia(nombres, bases, anchoBase, completa), crea secuencia con todos sus atributos
- CantidadBases(c), calcula la cantidad de Bases que hay en total en una secuencia

TAD Genoma

Conjunto mínimo de datos:

- secuencias, lista de Secuencia, contiene todas las secuencias que conforman al genoma

Operaciones:

- Genoma(), crea un genoma vacío.
- Genoma(secuencias) crea un genoma con sus secuencias.
- AgregarSecuencia(s), añade a secuencias una secuencia correspondiente al genoma.
- SecuenciasCargadas(), busca la cantidad de secuencias del genoma.
- BuscarSecuencia(n), busca una secuencia dado su nombre.
- ExisteSub(sb), dada una subsecuencia se busca en el genoma y la cantidad de veces presente en él.
- Enmascarar(sbs), dada una subsecuencia la busca exacta en el genoma y enmascara la secuencia por 'X'
- sumarFrecuencia(base, a, c, g, t, u, menos, r, y, m, k, ss, w, b, d, h, v, n), asigna la cantidad de cada base presente en una secuencia
- Frecuencia(s), dada una secuencia se encarga de crear los indicadores de la cantidad de sus bases
- Codificar(nombreArchivo), codifica a lenguaje binario un genoma recibido en ASCII.
- Decodificar(nombreArchivo), decodifica un genoma de binario a ASCII

TAD ArchivoFasta

Conjunto mínimo de datos:

- nombreArchivo, cadena de caracteres, representa el nombre del archivo a leer o escribir

Operaciones:

- ArchivoFasta(), crear un archivo de tipo fasta vacío
- ArchivoFasta(nombreArchivo), crear un archivo de tipo fasta y asignándole un nombre
- Genoma CargarArchivo(nombreArchivo), carga la información de un genoma a partir de un archivo
- GuardarArchivo(g), guarda la información de un genoma en un archivo

TAD Nodo

Conjunto mínimo de datos:

- simbolo, carácter, representa al carácter de un genoma
- frecuencia, numero entero, indica la cantidad de veces que se presenta un simbolo en el genoma
- izquierdo, de tipo nodo, indica al nodo descendiente a su izquierda
- derecho, de tipo nodo, indica al nodo descendiente a su derecha

Operaciones:

- Nodo(sim, freq), se encarga de crear los nodos con los simbolos
- Nodo(izq, der), se encarga de crear los nodos que llevan a los simbolos
- esHoja(), reconoce si un nodo contiene o no un simbolo

TAD ArbolHuffman

Conjunto mínimo de datos:

- raiz, de tipo nodo, indica el primer nodo del arbol

Operaciones:

- ArbolHuffman(), crea un arbol con raiz
- ~ArbolHuffman(), destruye un arbol
- construirDesdeFrecuencias(frecuencias), crea el arbol a partir de los simbolos y sus frecuencias
- generarCodigos(tablaDeCodigos), crea el diccionario de codigos
- generarCodigosRec(codigoActual, tablaDeCodigos), se encarga de asegurar que se cree el diccionario para cada caracter
- destruirArbol(nodo), borra los nodos en un arbol

TAD Codificador

Conjunto mínimo de datos:

Operaciones:

- codificar(texto, tablaDeCodigos), se encarga de convertir el genoma a lenguaje binario
- Decodificar(datos, longitudOriginal, raiz), se encarga de convertir el genoma a lenguaje ASCII

TAD Comando

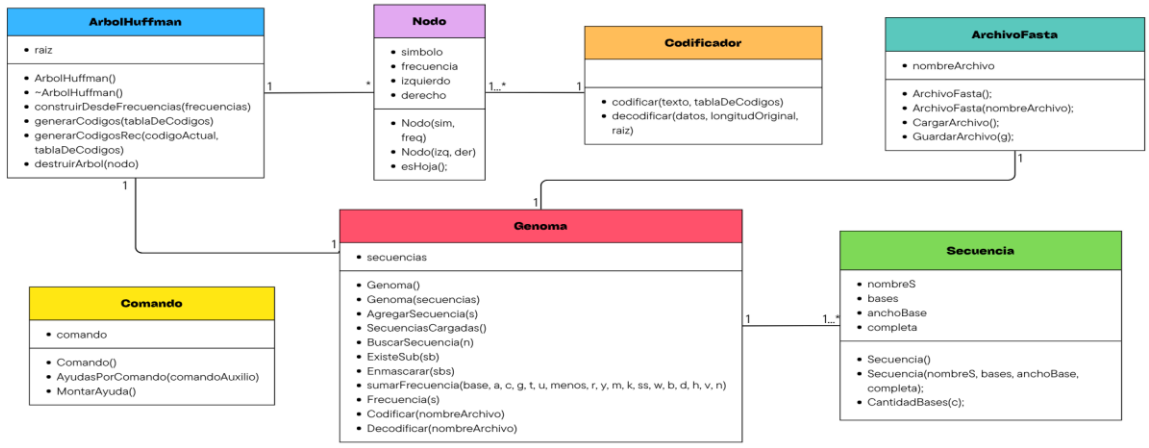
Conjunto mínimo de datos:

- comando, cadena de caracteres, recibe la opción que el usuario ingresa

Operaciones:

- Comando(), crea un comando vacío.
- AyudasPorComando(comandoAuxilio), muestra la descripción de cada comando
- MontarAyuda(), muestra todos los comandos disponibles

Diagrama de TADs



Flujo de operaciones

1) cargar <archivo.fasta>

1. Inicio
2. Abrir archivo nombre_archivo
3. ¿Archivo abierto?
    a. No → Mostrar: "<nombre_archivo> no se encuentra o no puede leerse" → Fin
    b. Sí → Inicializar Genoma y variables
4. Leer línea por línea
5. ¿Línea comienza con >?
    a. Sí:
        i. ¿Existe secuencia previa?
            1. Sí → Guardar secuencia en Genoma
            2. No → Iniciar nueva secuencia con nombre
        ii. Continuar leyendo líneas
    b. No → Procesar caracteres: quitar espacios, actualizar ancho, comprobar '-', agregar a secuencia
6. Repetir paso 4
7. ¿Fin de archivo?
    a. No → Repetir lectura
    b. Sí:
        i. ¿Hay secuencia pendiente?
            1. Sí → Guardar secuencia en Genoma

8. Cantidad de secuencias:
    a. 0 → Mostrar: "<nombre_archivo> no contiene ninguna secuencia."
    b. 1 → Mostrar: "1 secuencia cargada correctamente desde <nombre_archivo>"
    c. 1 → Mostrar: "N secuencias cargadas correctamente desde <nombre_archivo>"
9. Fin

2) guardar <archivo.fasta>

1. Usuario ejecuta guardar archivo.fasta
2. ArchivoFasta recibe nombreArchivo
3. Abrir archivo con ofstream
4. ¿Archivo abierto?
    a. No → Error: no se puede escribir archivo
    b. Sí → Recorrer todas las Secuencias en Genoma
5. Por cada secuencia:
    a. Escribir encabezado: >nombreSecuencia
    b. Escribir bases en líneas de anchoBase
6. Finalizar → Mensaje: genoma guardado

3) descripcion_secuencias

1. Usuario ejecuta descripcion_secuencias
2. ¿Genoma vacío?
    a. Sí → Mensaje: no hay secuencias cargadas
    b. No → Recorrer cada Secuencia en Genoma
3. Por cada secuencia:
    a. Imprimir nombre, cantidad de bases y estado "completa"
4. Fin comando

4) histograma <nombre_secuencia>

1. Usuario ejecuta histograma <nombre>
2. Buscar secuencia en Genoma
3. ¿Encontrada?
    a. No → Mensaje: secuencia no existe
    b. Sí:
        i. Inicializar contadores A, C, G, T, U en 0
        ii. Recorrer cada base de la Secuencia
        iii. Aplicar tabla de equivalencias (R, Y, N, etc.)
        iv. Sumar al/los contadores correspondientes
4. Al finalizar → Imprimir frecuencias de A, C, G, T, U

5) buscar_sub <subsecuencia>

1. Usuario ejecuta buscar_sub <subsecuencia>
2. ¿Genoma vacío?
    a. Sí → Mensaje: no hay secuencias cargadas
    b. No → Recorrer cada Secuencia
3. Comparar bases con subsecuencia carácter a carácter
4. ¿Coincidencia completa?

    a. Sí → Incrementar contador de apariciones
    b. No → Continuar comparando
5. Al finalizar cada secuencia → Pasar a la siguiente
6. Al finalizar todas → Imprimir cantidad total de apariciones
7.

6) enmascarar <subsecuencia>

1. Usuario ejecuta enmascarar <subsecuencia>
2. ¿Genoma vacío?
    a. Sí → Mensaje: no hay secuencias cargadas
    b. No → Recorrer cada Secuencia (modificable)
3. Buscar coincidencia carácter a carácter
4. ¿Coincidencia completa?
    a. Sí → Retroceder x posiciones con iterador
        i. Reemplazar caracteres por 'N'
    b. No → Continuar
5. Al finalizar cada secuencia → Pasar a la siguiente
6. Al finalizar todas:
    a. ¿Se modificó alguna?
        i. Sí → Mensaje: subsecuencia enmascarada
        ii. No → Mensaje: no encontrado

7) codificar <archivo.fabin>

1. Usuario ejecuta codificar <archivo.fabin>
2. ¿Genoma vacío? a. Sí → Mensaje: no hay secuencias cargadas b. No → Continuar
3. Calcular frecuencia de todas las bases en todas las secuencias
4. Construir Árbol de Huffman a partir de las frecuencias
5. Generar tabla de códigos binarios para cada base
6. Abrir archivo.fabin para escritura binaria
7. ¿Archivo abierto con éxito? a. No → Mensaje: Error guardando el archivo b. Sí → Escribir cabecera (tabla de frecuencias, N° de secuencias)
8. Para cada Secuencia en Genoma:
    a. Escribir metadatos de la secuencia (nombre, longitud, etc.)
    b. Codificar bases a bytes y escribirlos en el archivo
9. Al finalizar el bucle → Cerrar archivo
10. Mensaje: Secuencias codificadas y guardadas

8) decodificar <archivo.fabin>

1. Usuario ejecuta decodificar <archivo.fabin>
2. Abrir archivo.fabin para lectura binaria
3. ¿Archivo abierto con éxito? a. No → Mensaje: Error leyendo el archivo b. Sí → Limpiar/sobrescribir secuencias actuales en memoria
4. Leer cabecera del archivo (tabla de frecuencias)
5. Reconstruir Árbol de Huffman
6. Leer número total de secuencias (N)
7. Bucle N veces:
    a. Leer metadatos de la siguiente secuencia

    b. Leer los bytes de datos codificados
    c. Decodificar los bytes a bases usando el Árbol de Huffman
    d. Crear y guardar la nueva Secuencia en memoria
8. Al finalizar el bucle → Cerrar archivo
9. Mensaje: Secuencias decodificadas y cargadas

9) salir

1. Usuario ejecuta salir
2. Mostrar mensaje de salida
3. Finalizar ejecución del programa

Plan de Pruebas – Decodificar

Caso	Entrada	Resultado esperado
Intenta decodificar archivo de genoma no existente	decodificar g	(mensaje de error) No se pueden cargar las secuencias desde g.fabin
Intenta decodificar un archivo de genoma bien codificado	decodificar archivoGuardar	(decodificación exitosa) Secuencias decodificadas desde archivoGuardar.fabin y cargadas en memoria.
Intenta decodificar un archivo que no es .fabin	Decodificar g	(mensaje de error) No se pueden cargar las secuencias desde g.fabin