UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

ESCUELA DE INGENIERIA

AUXILIA: GUILLERMO FRANCISCO MELLINI SALGUERO

PRACTICA 1

INVENTARIO DE EQUIPO

MANUAL TECNICO

PRESENTADO POR

DAVID ORLANDO FUENTES MORALES

REGISTRO ACADÉMICO 201709022

DPI 2998989540101

GUATEMALA, 22 DE AGOSTO DE 2024

TABLA DE CONTENIDO

INTRODUCCION

El presente documento describe el diseño de un programa creado con fortran, el cual su utilización es la creación, modificación de inventarios de equipos, y a su vez la creación de archivos .txt como su informe del inventario.

Propósito del documento

El propósito es proporcionar una guía del uso y explicación de la creación de inventarios utilizando código fortran,

Describir los métodos y técnicas de prueba utilizadas para la lectura de documentos y creación de archivos txt y funcionalidad del código, incluyendo casos de prueba, resultados esperados y procedimientos de validación.

Código

Explicación

- Se comienza importando módulos creados para el funcionamiento del programa. Se declaran variables que se usaran para almacenar la opción del menú y rutas de archivos a utilizar por el usuario.

```
program Inventario

    use InventarioModule
    use LeerArchi
    use Movimientos_inventario_module
    use Crear_Informe
    implicit none
    integer :: opcion_menu, max_lineas, i
    character(len=100) :: contenido(100)
    character(len=100) :: ruta_archivo, ruta_instrucciones,ruta_archivo_editado,ruta_archivo_movimientos
    character(len=100) :: delimitador
    type(producto),allocatable :: productos(:)
    i=1
```

- Se inicializan las variables.

```
    max_lineas = 100
    !ruta_archivo="inventario.inv"
    delimitador=","
    ruta_archivo=""
    ruta_instrucciones=""
    ruta_archivo_editado=""
    ruta_archivo_movimientos=""
    allocate(productos(0))
```

- Se usa un bucle Do para la creación del menú y su utilización sea amigable con el usuario.

```
!call LeerArchivo(max_lineas,contenido,ruta_archivo)



do
    print *, "Practica 1: Inventario de productos"
    print *, "Menu de opciones de inventario de productos"
    print *, "1. Cargar Inventario Inicial"
    print *, "2. Cargar Instrucciones de Movimientos"
    print *, "3. Crear Informe de Inventario"
    print *, "4. Salir"
     print *, "Ingrese una opcion: "
     read *, opcion_menu
      ruta_archivo_editado=ruta_archivo
        ruta_archivo_movimientos=ruta_instrucciones

        !  ruta_archivo=""
        !  ruta_instrucciones=""

     select case(opcion_menu)
        case(1)
           !call LeerArchivo(max_lineas,contenido,ruta_archivo)
            !do i=1, max_lineas

            call Cargarinventario(ruta_archivo)
            ruta_archivo_editado=ruta_archivo
            !end do
        case(2)
            !call LeerArchivo(max_lineas,contenido,ruta_archivo)
            call cargar_movimientos(ruta_archivo_editado,ruta_instrucciones)
            ruta_archivo_movimientos=ruta_instrucciones
            !ruta_instrucciones=""

        case(3)
            call Crearinforme(ruta_archivo,ruta_archivo_movimientos,ruta_archivo_editado)
        case(4)
            exit
        case default
            print *, "Opcion invalida"
    end select
end do
```

- Dentro de cada opción del bucle llamamos a cada subroutine creada específicamente para realizar dicha opción seleccionada por el usuario.
- Opción 1: Cargarinventario carga el inventario inicial desde un archivo.
- Opción 2: cargar_movimientos carga las instrucciones de movimientos de inventario.
- Opción 3: Crearinforme crea un informe del inventario basado en los archivos cargados.
- Opción 4: Sale del programa.

```
select case(opcion_menu)
   case(1)
      !call LeerArchivo(max_lineas,contenido,ruta_archivo)
       !do i=1, max_lineas

      call Cargarinventario(ruta_archivo)
      ruta_archivo_editado=ruta_archivo
      !end do
   case(2)
       !call LeerArchivo(max_lineas,contenido,ruta_archivo)
      call cargar_movimientos(ruta_archivo_editado,ruta_instrucciones)
      ruta_archivo_movimientos=ruta_instrucciones
      !ruta_instrucciones=""

   case(3)
       call Crearinforme(ruta_archivo,ruta_archivo_movimientos,ruta_archivo_editado)
   case(4)
       exit
   case default
       print *, "Opcion invalida"
end select
```

- Se crea un modulo el cual contendrá subroutinas,
- La subrutina Cargar inventario se encarga de leer el archivo.inv y luego extraer su información y la almacena para su uso

```
subroutine Cargarinventario(ruta_archivo)
    implicit none

    character(len=100), intent(inout) :: ruta_archivo
    character(len=100) :: linea, palabra1, palabra2, palabra3, palabra4, palabra5
    integer:: unit, ios,pos1,pos2,pos3,pos4,pos5
    integer,dimension(100)::cantidad
    real,dimension(100)::precio
    character(len=100),dimension(100):: equipo,ubicacion
    integer:: contador,i
    !inicializar el contador
    !nombre=""
    unit=10
    contador=0
    !abrir el archivo
    print*, "Ingrese el nombre del archivo o la ruta"
    read(*,"(A)") ruta_archivo
    open(unit=unit,file=trim(ruta_archivo),status='old',action='read',iostat=ios)
    if(ios/=0) then
        print*, "Error al abrir al abrir el inventario: ", trim(ruta_archivo)
        return
    end if
    !leer el archivo y almacenar los datos en el arreglo productos
    do
        read(10,'(A)',iostat=ios) linea
        if(ios/=0) exit

        !separar la linea en palabras
        pos1=index(linea," ")
        pos2=index(linea(pos1+1:),";")+pos1
        pos3=index(linea(pos2+1:),";")+pos2
        pos4=index(linea(pos3+1:),";")+pos3

        !extraer las palabras
        palabra1=linea(1:pos1-1)
        palabra2=linea(pos1+1:pos2-1)
        palabra3=linea(pos2+1:pos3-1)
        palabra4=linea(pos3+1:pos4-1)
        palabra5=linea(pos4+1:)
        !almacenar los datos en el arreglo productos
        contador=contador+1
        equipo(contador)=trim(palabra2)
        read(palabra3,*) cantidad(contador)
        read(palabra4,*) precio(contador)
        ubicacion(contador)=trim(palabra5)
    end do
    close(10)
```

- Se declaran variables y se inicializan

```
character(len=100), intent(inout) :: ruta_archivo
character(len=100) :: linea, palabra1, palabra2, palabra3, palabra4, palabra5
integer:: unit, ios,pos1,pos2,pos3,pos4,pos5
integer,dimension(100)::cantidad
real,dimension(100)::precio
character(len=100),dimension(100):: equipo,ubicacion
integer:: contador,i
!inicializar el contador
!nombre=""
unit=10
contador=0
!abrir el archivo
```

- Al usuario se le pedirá que ingrese las rutas de los archivos a leer.
- El código que se utiliza para abrir los archivos es el siguiente:

```
print*, "Ingrese el nombre del archivo o la ruta"
read(*,"(A)") ruta_archivo
open(unit=unit,file=trim(ruta_archivo),status='old',action='read',iostat=ios)
if(ios/=0) then
    print*, "Error al abrir al abrir el inventario: ", trim(ruta_archivo)
    return
end if
!leer el archivo y almacenar los datos en el arreglo productos
do
```

- Se crea un bucle do lee el archivo línea por línea. Cada línea se separa en palabras usando la función index para encontrar los delimitadores (;). Las palabras extraídas se almacenan en los arreglos equipo, cantidad, precio y ubicación. Al final imprime cada línea del archivo.

```
!leer el archivo y almacenar los datos en el arreglo productos
do
    read(10,'(A)',iostat=ios) linea
    if(ios/=0) exit

    !separar la linea en palabras
    pos1=index(linea," ")
    pos2=index(linea(pos1+1:),";")+pos1
    pos3=index(linea(pos2+1:),";")+pos2
    pos4=index(linea(pos3+1:),";")+pos3

    !extraer las palabras
    palabra1=linea(1:pos1-1)
    palabra2=linea(pos1+1:pos2-1)
    palabra3=linea(pos2+1:pos3-1)
    palabra4=linea(pos3+1:pos4-1)
    palabra5=linea(pos4+1:)
    !almacenar los datos en el arreglo productos
    contador=contador+1
    equipo(contador)=trim(palabra2)
    read(palabra3,*) cantidad(contador)
    read(palabra4,*) precio(contador)
    ubicacion(contador)=trim(palabra5)
end do
close(10)
!mostrar los datos almacenados
print*, "Datos almacenados en el arreglo productos"
print*, "Nombre, Cantidad, Precio, Ubicacion"
do i=1,contador
```

```
subroutine Cargarinventario(ruta_archivo)
    implicit none

    character(len=100), intent(inout) :: ruta_archivo
    character(len=100) :: linea, palabra1, palabra2, palabra3, palabra4, palabra5
    integer:: unit, ios,pos1,pos2,pos3,pos4,pos5
    integer,dimension(100)::cantidad
    real,dimension(100)::precio
    character(len=100),dimension(100):: equipo,ubicacion
    integer:: contador,i
    !inicializar el contador
    !nombre=""
    unit=10
    contador=0
    !abrir el archivo
    print*, "Ingrese el nombre del archivo o la ruta"
    read(*,"(A)") ruta_archivo
    open(unit=unit,file=trim(ruta_archivo),status='old',action='read',iostat=ios)
    if(ios/=0) then
        print*, "Error al abrir al abrir el inventario: ", trim(ruta_archivo)
        return
    end if
    !leer el archivo y almacenar los datos en el arreglo productos
    do
        read(10,'(A)',iostat=ios) linea
        if(ios/=0) exit

        !separar la linea en palabras
        pos1=index(linea," ")
        pos2=index(linea(pos1+1:),";")+pos1
        pos3=index(linea(pos2+1:),";")+pos2
        pos4=index(linea(pos3+1:),";")+pos3

        !extraer las palabras
        palabra1=linea(1:pos1-1)
        palabra2=linea(pos1+1:pos2-1)
        palabra3=linea(pos2+1:pos3-1)
        palabra4=linea(pos3+1:pos4-1)
        palabra5=linea(pos4+1:)
        !almacenar los datos en el arreglo productos
        contador=contador+1
        equipo(contador)=trim(palabra2)
        read(palabra3,*) cantidad(contador)
        read(palabra4,*) precio(contador)
        ubicacion(contador)=trim(palabra5)
    end do
    close(10)
```

- Subroutine Cargar movimientos
- En esta parte del código se leerá el contenido extraído del archivo.txt y luego se correrán acciones en ella, como elimina y agregar a stock.
- Para agregar a stock utilizamos if el cual nos ayudara a validar que acción se debe hacer.
- Dentro de if utilizamos un ciclo do el cual nos ayudara a validar la acción de agregar las unidades de equipo.

```
if(trim(palabra1)== "agregar_stock") then
    !buscar el producto en el arreglo productos
    do i=1,contador
        if(trim(equipo(i))==trim(palabra2) .and. trim(ubicacion(i))==trim(palabra5)) then
            read(palabra3,*) valornumerico
             !actualizar el cantidad
             cantidad(i)=cantidad(i)+valornumerico
             print*, "Se agrego ", trim(palabra3), " unidades de ", trim(palabra2)
             exit
        end if
end do
```

- Para eliminar stock , validamos la acción con if y luego de ello utilizamos un do el cual ayudara a actualizar la cantidad de equipo que quedara si la acción es valida.

```
else if(trim(palabra1)== "eliminar_equipo") then
        !buscar el producto en el arreglo productos
        do i=1,contador
            if(trim(equipo(i))==trim((palabra2)).and.trim(ubicacion(i))==trim(palabra5)) then
                !verificar si la cantidad a eliminar es mayor a la cantidad en inventario
                read(palabra3,*) valornumerico
               if(cantidad(i)>=valornumerico) then

                   cantidad(i)=cantidad(i)-valornumerico
                   print*, "Se elimino ", trim(palabra3), " unidades de ", trim(palabra2)

                else
                    print*,"No hay suficiente inventario para eliminar ", trim(palabra3), " unidades de ", trim(palabra2)
                end if
                exit
            end if
        end do
    if ( trim(ubicacion(i))/=trim(palabra5)) then
        print *, 'Error: El equipo', trim(palabra2), 'no existe en la ubicación', trim(palabra5)
    end if
end if
```

- Utilizamos para aumentar o disminuir el stock con suma y resta de los valores y también usamos validar una variable tipo carácter y utilizarla para las operaciones.

```
read(palabra3,*) valornumerico
if(cantidad(i)>=valornumerico) then

   cantidad(i)=cantidad(i)-valornumerico
   print*, "Se elimino ", trim(palabra3), " unidades de ", trim(palabra2)

else
    print*,"No hay suficiente inventario para eliminar ", trim(palabra3), " unidades de ", trim(palabra2)
end if
exit
```

- Para la creación del informe
    - Se le pide al usuario con que nombre guardara el informe.
    - Se imprime en consola el informe y se crea un archivo

```
! Solicita al usuario ingresar la ruta del archivo de informe
print *, 'Ingrese la ruta del archivo de informe (.txt):'
read(*, '(A)') ruta_archivo_editado
 !guardar el inventario actualizado en el archivo de inventario
  open(unit=12, file=trim(ruta_archivo_editado), status='replace', action='write', iostat=ios)
      if(ios/=0) then
         print*, "Error al abrir al abrir el archivo de movimientos: ", trim(ruta_archivo)
         return
      end if
      write(12,'(A)') "Informe de inventario"
      write(12,'(A)') "Equipo  Cantidad   Precio Unitario    valor    Ubicacion"
      write(12,'(A)') "-----------------------------------------------------"
      print*, "Informe de inventario"
      print*, "Equipo    Cantidad    Precio_Unitario     valor       Ubicacion"
      print*, "-----------------------------------------------------------------"
```

- En el archivo creado del informe utilizamos un write y vamos modificando su posición
    - A: Escribe una cadena de caracteres.
    - 2X: Añade dos espacios en blanco.
    - I5: Escribe un entero en un campo de 5 caracteres.
    - 7X: Añade siete espacios en blanco.
    - F10.2: Escribe un número real en un campo de 10 caracteres con 2 decimales

```
    do i=1,contador
      total=cantidad(i)*precio(i)
      signo="Q"
        write(12,'(A,2X,I5,7X,F10.2,5X,A,F10.2,2x,A)') trim(equipo(i)), cantidad(i), precio(i),"Q",total, trim(ubicacion(i))
        print*, trim(equipo(i)), cantidad(i), precio(i),signo,cantidad(i)*precio(i), ubicacion(i)
    end do
   print*, "Inventario actualizado guardado en el archivo ", trim(ruta_archivo_editado)
  print*, "Fin de la actualizacion de inventario"
  close(12)
  end subroutine Crearinforme
```