Esquema	Que Se Recorre?	Que Quiero Hacer?

• Procesamiento
• Búsqueda
  • Sabiendo que esta
  • Desconociendo si esta
• Acumulacion
• Máximos Y Mínimos
	• Celdas Del Tablero
	• Celdas De Una Fila
	• Celdas De Una Columna
	• Filas Del Tablero
	• Columnas Del Tablero
	• Enumerativos:
	  • Colores
	  • Direcciones
		• Poner, Copiar……..

Formato general del contrato Observación

Observación: Es Un Recorrido……Sobre……Haciendo…….

Ejemplo:

Observación: Es un Recorrido *de Procesamiento* sobre *Las Celdas de una columna* (que quiero hacer) *Borrando las bolitas Rojas*

<u>Esquema General De Recorrido</u>

```
Function – procedure NombreGenerico () {
    Iniciar recorrido
    while (quedan elementos para recorrer) {
        Procesar elemento
        Avanzar al siguiente elemento
    }
Finalizar recorrido
}
```

Pasos para la implementación del recorrido:

1. Que voy a recorrer?
2. Que Esquema voy a utilizar?
3. Que quiero hacer?

Ejemplo:

Que Voy a Recorrer?

Celdas del tablero

Que esquema Voy a utilizar?

Procesamiento

Que quiero Hacer?

Sacar todas las bolitas de color rojo

Contrato:

/* Observación : Es un recorrido de **procesamiento** sobre **celdas del tablero** , **quitando todas las bolitas de color rojo**

Estructuras genéricas de Recorridos
Que Se Recorre?

```
################## RECORRIDO POR CELDA DE  UN TABLERO #################

procedure RecorridoPorCeldasDeUnTableroGenericoConParametros (dirPrincipal, dirSecundaria) {
   /*
   Observaciones: Es un recorrido de (Esquema) por las celdas o columna del de una fila
   */

         IrAlInicioEnUnRecorridoPorCeldasAlyAl_( dirPrincipal, dirSecundaria)
         while (haySiguienteCeldaEnUnRecorridoPorCeldasAl_( dirPrincipal)) {


          IrASiguienteCeldaEnUnRecorridoPorCeldasAlyAl_( dirPrincipal, dirSecundaria)
         }
}

################################### RECORRIDO POR CELDA DE FILA ################################

procedure RecorridoPorFilaDeUnTablero (dirAMover) {
   /*
      Observaciones: Es un recorrido de (Esquema) por las celdas del tablero por Fila de Oeste-Este
   */
   IrALaPrimerFilaEnRecorridoPorFilaAl_(dirAMover)
   while (haySiguienteFilaEnUnRecorridoPorFilaAl_(dirAMover)) {


   IrASiguienteFilaEnUnRecorridoPorFilaAl_(dirAMover)
   }
}
 ################################### RECORRIDO POR COLUMNA DE TABLERO ######################

procedure RecorridoPorColumnaDeUnTablero (dirAMover) {
/*
Observaciones: Es un recorrido de (Esquema) por las celdas del tablero por Columna Norte-Sur haciendo
*/
   IrALaPrimerColumnaEnRecorridoPorFilaAl_(dirAMover)
   while (haySiguienteColumnaEnUnRecorridoPorFilaAl_(dirAMover)) {


   IrASiguienteColumnaEnUnRecorridoPorColumnaAl_(dirAMover)
   }
}
```

```
################## RECORRIDO ENUMERATIVO DE DIRECCION Y COLOR ##################

procedure RecorridoPorEnumerativoDeDireccion(){
   /*
   Observaciones: Es un recorrido de ...(completar) por "direcciones" haciendo ...(completar)
   */
   direccionActual := minDir()

   while (direccionActual /= maxDir()){
      direccionActual:= siguiente(direccionAcual)
   }
   minDir() Norte > Este > Sur > Oeste maxDir()
   maxDir() Oeste > Sur > Este > Norte minDir()
}
################## RECORRIDO ENUMERATIVO DE DIRECCION Y COLOR ##################

procedure RecorridoPorEnumerativoDeColor(){
   /*
   Observaciones: Es un recorrido de ...(completar) por "colores" haciendo ...(completar)
   */
   colorActual := minColor()

   while (direccionActual /= maxColor()){
      colorActual:= siguiente(colorActual)
   }
         minColor() Azul > Negro > Rojo > Verde maxColor()
          maxColor() Verde > Rojo > Negro > Azul minColor()
}
```

Estructura genérica de recorridos II

Como se Recorre?

PROCESAMIENTO

```
################## ESQUEMA DE PROCESAMIENTO POR CELDAS DEL TABLERO ##################

procedure RecorridoDeProcesamientoPorCeldasDelTablero (primerDireccion, segundaDireccion) {
   /*
      Observaciones: Es un recorrido de procesamiento por celdas del tablero
      haciendo ...(lo que haya que hacer - ProcesarCeldaActual())
   */
   IrAlInicioEnUnRecorridoAl_YAl_(primerDireccion, segundaDireccion)
   while (haySiguienteCeldaEnUnRecorridoAl_YAl_(primerDireccion, segundaDireccion)) {
      ProcesarCeldaActual()
      IrASiguienteCeldaEnUnRecorridoAl_YAl_(primerDireccion, segundaDireccion)
   }
   ProcesarCeldaActual() // CASO BORDE
}

############## ESQUEMA DE PROCESAMIENTO POR DIRECCIONES (ENUMERATIVOS) ##################

procedure RecorridoPorEnumerativoDeDireccion(){
   /*
   Observaciones: Es un recorrido de procesamiento por direcciones del tablero
      haciendo ...(lo que haya que hacer - ProcesarDireccionActual())
   */
   direccionActual := minDir()                          //Se inicia el recorrido iniciando la variable/

   while (direccionActual /= maxDir()){
      ProcesarDireccionActual()
      direccionActual:= siguiente(direccionAcual)

   }
   ProcesarDireccionActual() // CASO BORDE
   // minDir() Norte > Este > Sur > Oeste maxDir()
   // maxDir() Oeste > Sur > Este > Norte minDir()
}
```

```
######### ESQUEMA DE PROCESAMIENTO POR COLOR (ENUMERATIVOS) ######################

procedure RecorridoPorEnumerativoDeColor(){
   /*
   Observaciones: Es un recorrido de procesamiento por colores del tablero
      haciendo ...(lo que haya que hacer - ProcesarColorActual())
   */
   colorActual := minColor()

   while (direccionActual /= maxColor()){
      ProcesarColorActual()
      colorActual:= siguiente(colorActual)
   }
   ProcesarColorActual() // CASO BORDE

   // minColor() Azul > Negro > Rojo > Verde maxColor()
   // maxColor() Verde > Rojo > Negro > Azul minColor()
}
```

BUSQUEDA SABIENDO QUE ESTA O NO

```
####### ESQUEMA DE RECORRIDO DE BUSQUEDA (SABIENDO QUE ESTA )EN CELDA DE TABLERO ############

procedure RecorridoPorCeldasGenericoConParametros (primerDireccion, segundaDireccion) {
  /*
    Observaciones:
    Es un recorrido de "busqueda" sabiendo que esta lo que busco por las celdas del tablero
    buscando ... ( lo que tenga que buscar )
    PRECONDICION:
      Hay lo que estoy buscando, si no llega a estar "BOOM".
  */

  IrAlInicioEnUnRecorridoAl_YAl_(Este, Norte)
  while (not (estaLoQueEstoyBuscando())) {  //Como se que esta lo que busco no evaluo el caso de si hay prox celda
    IrASiguienteCeldaEnUnRecorridoAl_YAl_(Este, Norte)
  }

}

####### ESQUEMA DE RECORRIDO DE BUSQUEDA (NO SABIENDO SI ESTA) EN CELDA DE TABLERO ############

procedure RecorridoPorCeldasGenericoConParametros (primerDireccion, segundaDireccion) {
  /*
    Observaciones:
    Es un recorrido de "busqueda" no sabiendo si esta lo que busco por las celdas del tablero
    buscando ... ( lo que tenga que buscar )
    PRECONDICION:  No posee en este caso se vuelve total
  */

  IrAlInicioEnUnRecorridoAl_YAl_(Este, Norte)
  while ( haySiguienteCeldaEnUnRecorridoAl_YAl_(primerDireccion, segundaDireccion) && (not
(estaLoQueEstoyBuscando())) {  //Como no se si esta salvo el caso
    IrASiguienteCeldaEnUnRecorridoAl_YAl_(Este, Norte)
  }

}
```

```
##ESQUEMA DE RECORRIDO DE BUSQUEDA (NO SABIENDO SI ESTA ) Por Direcciones (Enumerativo) ######

procedure RecorridoPorCeldasGenericoConParametros (primerDireccion, segundaDireccion) {
   /*
      Observaciones:
      Es un recorrido de "busqueda" no sabiendo si esta lo que busco por direcciones
      buscando ... ( lo que tenga que buscar )
      PRECONDICION:  No posee en este caso se vuelve total
   */

   direccionActual:= minDir()
   while (direccionActual /= maxDir() && not (estaLoQueEstoyBuscando() ){
         direccionActual:= siguiente(direccionActual)
   }


   // minColor() Azul > Negro > Rojo > Verde maxColor()
   // maxColor() Verde > Rojo > Negro > Azul minColor()
}
```

ACUMULACION

```
################## ESQUEMA DE RECORRIDO DE ACUMULACION POR CELDAS DEL TABLERO ##################

procedure RecorridoDeAcumulacionPorCeldasDelTablero (primerDireccion, segundaDireccion) {
   /*
      Observaciones: Es un recorrido de Acumulacion por celdas del tablero
      haciendo ...(lo que haya que hacer - ProcesarCeldaActual())
   */
   IrAlInicioEnUnRecorridoAl_YAl_(primerDireccion, segundaDireccion)
   totalAcumulado := cantidadAContarAquí()                    // Se inicializa el acumulador
   while (haySiguienteCeldaEnUnRecorridoAl_YAl_(primerDireccion, segundaDireccion)) {
      totalAcumulado := totalAcumulado + cantidadAContarAquí()
      IrASiguienteCeldaEnUnRecorridoAl_YAl_(primerDireccion, segundaDireccion)
   }
 }
################## ESQUEMA DE RECORRIDO DE ACUMULACION POR DIRECCIONES  ##################

procedure RecorridoDeAcumulacionPorAcumulacion () {
   /*
      Observaciones: Es un recorrido de Acumulacion por direcciones
      haciendo ...(lo que haya que hacer - ProcesarCeldaActual())
   */
   direccionActual:= minDir()
  totalAcumulado := cantidadAContarAquí()
   while (direccionActual /= maxDir()){
            totalAcumulado := totalAcumulado + cantidadAContarAquí()
           direccionActual:= siguiente(direccionActual)
   }
   // minColor() Azul > Negro > Rojo > Verde maxColor()
   // maxColor() Verde > Rojo > Negro > Azul minColor()
}
```

CantidadAContarHacia_() es una subtarea que dada una dirección, describe la cantidad de elementos que se cuentan hacia dicha dirección

MAXIMOS Y MINIMOS

```
################## ESQUEMA DE RECORRIDO DE MAXIMOS Y MINIMOS POR CELDAS DEL TABLERO ##########

procedure RecorridoDeMaximosYMinimosPorCeldasDelTablero (primerDireccion, segundaDireccion) {
  /*
    Observaciones: Es un recorrido de Maximos y minimos por celdas del tablero
    haciendo ...(lo que haya que hacer - ProcesarCeldaActual())
  */
  IrAlInicioEnUnRecorridoAl_YAl_(primerDireccion, segundaDireccion)
  minimoOMaximoAlMomento := cantidadAContarAquí()  // Se inicializa el elemento a comparar
  while (haySiguienteCeldaEnUnRecorridoAl_YAl_(primerDireccion, segundaDireccion)) {
    minimoOMaximoAlMomento := mínimoOMáximoEntre_Y_(minimoOMaximoAlMomento,
                                        cantidadAContarAquí()) // Se compara y tal vez se actualiza la variable
    IrASiguienteCeldaEnUnRecorridoAl_YAl_(primerDireccion, segundaDireccion)
  }
}

################## ESQUEMA DE RECORRIDO DE MAXIMOS Y MINIMOS POR DIRECCIONES ##########

procedure RecorridoDeMaximosYMinimosPorDirecciones() {
  /*
    Observaciones: Es un recorrido de Maximos y minimos por direcciones
    haciendo ...(lo que haya que hacer - ProcesarCeldaActual())
  */
  direccionActual:= minDir()
  minimoOMaximoAlMomento := cantidadAContarAquí()  // Se inicializa el elemento a comparar
  while (direccionActual /= maxDir()){
        minimoOMaximoAlMomento := mínimoOMáximoEntre_Y_(minimoOMaximoAlMomento,
                                        cantidadAContarAquí()) // Se compara y tal vez se actualiza la variable
        direccionActual:= siguiente(direccionActual)
  }
  // minColor() Azul > Negro > Rojo > Verde maxColor()
  // maxColor() Verde > Rojo > Negro > Azul minColor()
}
```

cantidadAContarAquí es una subtarea que describe el valor de los elementos que se cuentan en la celda actual.
mínimoOMáximoEntre_Y_ es una subtarea que dados dos valores describe entre ambos el más pequeño o más grande.
Notar que puede que en ocasiones no se desee el valor, sino algún otro criterio, como algo que identifique la celda de forma unívoca, su coordenada, etc.

IMPORTANTE
Un recorrido de máximos y mínimos es un recorrido que también busca cosas en el tablero, pero a diferencia del recorrido de búsqueda que se detiene cuando encuentra lo que busca, en este Esquema se busca en todo el tablero, Por ejemplo la celda que tiene mas cantidad de Bolitas, entender esto evita errores de implementación de recorrido