Capítulo 5 (WEB)

5.1.2 Ejercicios resueltos para la repetición do…while (Continuación…)

Ejercicio 5.1.2.3

Elaborar un algoritmo que proporcione el siguiente reporte:

CALIFICACIONES FINALES

NOMBRE	CAL.1	CAL.2	CAL.3	CALIF. FINAL	OBSERVACIÓN
XXXXXXXXXXXXXXXX	99.99	99.99	99.99	99.99	APROBADO
XXXXXXXXXXXXXXXX	99.99	99.99	99.99	99.99	REPROBADO
.	.	.	.	.	
.	.	.	.	.	
XXXXXXXXXXXXXXXX	99.99	99.99	99.99	99.99	REPROBADO

TOTAL APROBADOS: 999
TOTAL REPROBADOS: 999

A partir de que se tienen varios alumnos. Por cada alumno se tiene el NOMBRE, calificación 1,2, y 3, y que la calificación 1 vale el 20%, la 2 el 30% y la 3 el 50%. Deberá imprimir la observación APROBADO si la calificación final es mayor o igual a 60, en caso contrario imprimirá REPROBADO.

Al final se imprime el total de alumnos aprobados y el total de reprobados, es decir, debe contar la cantidad de alumnos que obtuvieron 60 o más de calificación, y la cantidad de alumnos que su calificación fue menor a 60.

(Primero hágalo usted, después compare la solución.)

```
Algoritmo ALUMNOS APROBADOS REPROBADOS
1. Declarar
     Variables
       nombreAlum: Cadena
       totAprobados, totReprobados: Entero
       calif1, calif2, calif3, caliFinal: Real
       observacion: Cadena
       desea: Carácter
2. Imprimir encabezado
3. totAprobados = 0
   totReprobados = 0
4. do
     a. Solicitar Nombre, calificación 1, calificación 2 y
                calificación 3
     b. Leer nombreAlum, calif1, calif2, calif3
     c. caliFinal = (calif1*0.2)+(calif2*0.3)+(calif3*0.5)
```

Con formato: Sangría: Izquierda: 0 cm

Con formato: Sangría: Izquierda: 0 cm

Con formato: Fuente: (Predeterminado) Arial, 12 pto

Con formato: Español (alfab. internacional)

```
      d. if caliFinal >= 60 then
            1. totAprobados = totAprobados + 1
            2. observacionobservación = "APROBADO"
      e. else
            1. totReprobados = totReprobados + 1
            2. observacionobservación = "REPROBADO"
      f. endif
      g. Imprimir nombreAlum, calif1, calif2,
                  calif3, caliFinal, observacionobservación
      h. Preguntar "¿Desea procesar otro alumno (S/N)?"
      i. Leer desea
5. while desea == 'S'
6. Imprimir totAprobados, totReprobados
7. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C505.C y
Programa en Java: Alumnos2.java

Explicación:

1. Se declaran las variables
2. Imprime encabezado
3. Inicia totAprobados en 0
   Inicia totReprobados en 0
4. Inicia el ciclo do
   a. Solicita los datos Nombre, calificación 1, 2 y 3
   b. Lee en nombreAlum, calif1, calif2, calif3
   c. Calcula caliFinal
   d. Si caliFinal >= 60 entonces
      1. Incrementa totAprobados en 1
      2. Coloca en observacionobservación "APROBADO"
   e. Si no (ELSE)
      1. Incrementa totReprobados en 1
      2. Coloca en observacionobservación "REPROBADO"
   f. Fin del IFif
   g. Imprime nombreAlum, calif1, calif2, calif3, caliFinal, observacionobservación
   h. Pregunta "¿Desea procesar otro alumno (S/N)?"
   i. Lee en desea la respuesta
5. Fin ciclo while desea == 'S',; Ssi se cumple, vuelve al do;, si no, sale del ciclo.
6. Imprime totAprobados, totReprobados
7. Fin del algoritmo

Ejercicio 5.1.2.4

Elaborar un algoritmo que mida la inflación que han tenido ciertos artículos y que imprima el siguiente reporte:

ANÁALISIS DE INFLACIÓON

ARTÍCULO	PRECIO ANTERIOR	PRECIO ACTUAL	PTJE. INFLACIÓON
XXXXXXXXXXX	99,999.99	99,999.99	99.99
XXXXXXXXXXX	99,999.99	99,999.99	99.99
.	.	.	.
.	.	.	.
XXXXXXXXXXX	99,999.99	99,999.99	99.99

Promedio de inflación: 99.99
Artículo con mayor inflación: XXXXXXXXXXXXXXXXXXXXXXXXXX
Porcentaje mayor de inflación: 99.99

Datos disponibles por cada artículo:
- Descripción
- Precio anterior
- Precio actual

Forma de calcular el porcentaje de inflación:

$$\text{PTJE. INFLACIÓON} = \frac{\text{Precio actual} - \text{Precio anterior}}{\text{Precio anterior}} \text{ X } 100$$

Se supone que los porcentajes de inflación de los artículos son diferentes.

(Primero hágalo usted, ...después compare la solución.)

```
Algoritmo INFLACIÓON DE ARTÍCULOS
1. Declarar
     Variables
       articulo, articuloMayor: Cadena
       totArticulos: Entero
       precioAct, precioAnt, ptjeInfla, totInfla,
       mayorInfla, promInfla: Real
       otro: Carácter
2. Imprimir encabezado
3. totArticulos = 0
   totInfla = 0
   mayorInfla = 0
4. do
     a. Solicitar Artículo, precio actual y precio anterior
     b. Leer articuloartículo, precioAct, precioAnt
     c. ptjeInfla = ((precioAct-precioAnt)/precioAnt)*100
     d. Imprimir articuloartículo, precioAnt, precioAct,
ptjeInfla
     e. if ptjeInfla > mayorInfla then
           1. mayorInfla = ptjeInfla
```

```
            2. articuloMayor = artículo
      f. endif
      g. totArticulos = totArticulos + 1
      h. totInfla = totInfla + ptjeInfla
      i. Preguntar "¿Desea procesar otro artíículo (S/N)?"
      j. Leer otro
5. while otro == 'S'
6. promInfla = totInfla / totArticulos
7. Imprimir promInfla, articuloMayor, mayorInfla
8. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C506.C y
Programa en Java: ArticulosInfla.java

Explicación:

1. Se declaran las variables
2. Imprime encabezado
3. Se inicia totArticulos en 0
   totInfla en 0
   mayorInfla en 0
4. Inicia el ciclo do
   a. Solicita los datos Artículo, precio actual y precio anterior
   b. Lee en ~~articulo~~artículo, precioAct, precioAnt
   c. Calcula el porcentaje de inflación ptjeInfla
   d. Imprime ~~articulo~~artículo, precioAnt, precioAct, ptjeInfla
   e. Si acaso ptjeInfla > mayorInfla entonces
      1. Coloca en mayorInfla el ptjeInfla
      2. Coloca en articuloMayor la descripción del artículo
   f. Fin del if
   g. Se incrementa totArticulos en 1
   h. Se incrementa totInfla con ptjeInfla
   i. Pregunta "¿Desea procesar otro artíículo (S/N)?"
   j. Lee en otro la respuesta
5. Fin del ciclo while otro == 'S',~~;~~ ~~S~~si se cumple, vuelve al do~~;~~, si no, sale del ciclo~~.~~
6. Calcula el promedio de inflación total promInfla
7. Imprime promInfla, articuloMayor, mayorInfla
8. Fin del algoritmo

Nota:
mayorInfla se inicia en 0 antes del ciclo, para que cuando entre la primera vez al ciclo y procese la inflación del primer artículo ptjeInfla y se compare en el if

```
      e. if ptjeInfla > mayorInfla then
            1. mayorInfla = ptjeInfla
            2. articuloMayor = artículo
      f. endif
```

Con formato: Justificado, Diseño: Claro (Gris 12,5%)

se cumpla la condición y entre al if a colocar los datos del primer artículo como el mayor porcentaje de inflación y el artículo del mayor porcentaje de inflación; y así, cuando deé la vuelta al ciclo y procese el segundo artículo, comparará el porcentaje de inflación de este segundo artículo, con el del primero, que es el que está como el mayor porcentaje, si se cumple la condición entrará al if a colocar los datos del segundo artículo como el mayor, y si no se cumple la condición dejará el artículo que está como el mayor, luego hará lo propio con el siguiente artículo y así hasta que se terminen los artículos.

Ejercicio 5.1.2.5

En una empresa manufacturera de mesas., Sse tienen varios obreros, y por cada obrero los datos:

Nombre del obrero: X----------X
Cada obrero puede haber trabajado varios días, por cada día que trabajó, se tiene un dato: Cantidad de unidades fabricadas (construidas).

Cantidad de unidades fabricadas: ---
Cantidad de unidades fabricadas: ---
.
Cantidad de unidades fabricadas: ---

Nota: cada cantidad fabricada es de un día de trabajo.

Nombre del obrero: X----------X
Cantidad de unidades fabricadas: ---
.
Cantidad de unidades fabricadas: ---

Elaborar un algoritmo que lea los datos de cada uno de los obreros e imprima el siguiente reporte:

REPORTE DE PRODUCCIÓON

NOMBRE DEL OBRERO	TOTAL PRODUCCIÓON
XXXXXXXXXXXXXXXXXXXXXXXX	999
XXXXXXXXXXXXXXXXXXXXXXXX	999
.	
.	
.	
XXXXXXXXXXXXXXXXXXXXXXXX	999
TOTAL 999 OBREROS	9999

Cálculos:
Se lee el nombre de un obrero, luego cada una de las cantidades producidas por día, se suman estas cantidades para calcular el TOTAL PRODUCCIÓN; en otras palabras, es la sumatoria de las cantidades producidas de todos los días que laboró.

Al final se pide el TOTAL de obreros y el TOTAL del TOTAL PRODUCCIÓN de todos los obreros. Además el nombre del obrero más productivo y la producción que fabricó, y el nombre del obrero menos productivo y la producción que fabricó. El obrero más productivo es el que tenga el TOTAL PRODUCCIÓN mayor. El obrero menos productivo es el que tenga el TOTAL PRODUCCIÓN menor. Se supone que el TOTAL PRODUCCIÓN de los obreros sones diferentes.

Con formato: Sangría: Primera línea: 1.25 cm

(Primero hágalo usted, ...después compare la solución.)

```
Algoritmo OBREROS
1. Declarar
     Variables
       nombreObr: Cadena
       proDia, totProd, toTotProd, totObreros: Entero
       otro, desea: Carácter
2. Imprimir encabezado
3. totObreros = 0
   toTotProd = 0
4. do
     a. Solicitar Nombre
     b. Leer nombreObr
     c. totProd = 0
     d. do
          1. Solicitar Producción del diadía
          2. Leer proDia
          3. totProd = totProd + proDia
          4. Preguntar "¿Desea procesar otro díia (S/N)?"
          5. Leer otro
     e. while otro == 'S'
     f. Imprimir nombreObr, totProd
     g. totObreros = totObreros + 1
        toTotProd = toTotProd + totProd
     h. Preguntar "¿Desea procesar otro obrero (S/N)?"
     i. Leer desea
5. while desea == 'S'
6. Imprimir totObreros, toTotProd
7. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C507.C y

Programa en Java: Obreros1.java

Explicación:

1. Se declaran las variables
2. Imprime encabezado
3. Se inician totObreros, toTotProd
4. Inicia el ciclo do
    a. Solicita el nombre
    b. Lee en nombreObr
    c. Inicia totProd en 0
    d. Inicia el ciclo do
        1. Solicita la Producción del ~~dia~~día
        2. Lee en proDia
        3. Incrementa totProd con proDia
        4. Pregunta "¿Desea procesar otro d~~í~~ía (S/N)?"
        5. Lee en otro la respuesta
    e. Fin del ciclo while otro == 'S' ~~;~~, ~~S~~si se cumple, vuelve al do~~;~~, si no, sale del ~~-~~ciclo~~.~~
    f. Imprime nombreObr, totProd
    g. Se incrementa totObr en 1
       Se incrementa toTotProd en totProd
    h. Pregunta "¿Desea procesar otro ~~obreo~~obrero (S/N)?"
    i. Lee en desea la respuesta
5. Fin del ciclo while desea == 'S' ~~;~~, ~~S~~si se cumple, vuelve al do~~;~~, si no, sale del ciclo.
6. Imprime totObreros, toTotProd
7. Fin del algoritmo

Nota:
Observe que este problema se resuelve utilizando dos ciclos do…while~~;~~, un primer ciclo que permite procesar varios obreros. Al procesar cada obrero, después de leer su nombre, se debe procesar varios días de trabajo, para lo cual, se utiliza un segundo ciclo do…while.

Además observe que no todos los contadores y acumuladores se inician en cero en el mismo lugar; antes del primer ciclo se inician los contadores y acumuladores que se imprimirán hasta el final del reporte, como son totObreros y toTotProd; los contadores y acumuladores que se usarán o imprimirán en el detalle del reporte como es el totProd, se inicia en cero antes del segundo ciclo (pero adentro del primer ciclo), que se incrementa adentro del segundo ciclo con proDia y que se imprime afuera del segundo ciclo, pero adentro del primer ciclo.

Con formato: Diseño: Claro (Gris 12,5%)

Ejercicio 5.1.2.6

Elaborar un algoritmo que lea los mismos datos del ejercicio anterior, pero ahora imprima el siguiente reporte:

REPORTE DE PRODUCCI~~O~~ÓN		
NOMBRE DEL OBRERO	TOTAL PRODUCCI~~O~~ÓN	SUELDO
XXXXXXXXXXXXXXXXXXXXXXX	999	99,999.99

XXXXXXXXXXXXXXXXXXXXXXX 999 99,999.99

.
.
.

XXXXXXXXXXXXXXXXXXXXXXX 999 99,999.99

TOTAL 999 OBREROS 9999 99,999.99

NOMBRE OBRERO ~~MAS~~MÁS PRODUCTIVO: XXXXXXXXXXXXXXXXXXXXX
PRODUCCIÓ~~O~~N QUE FABRICÓ~~O~~: 999

NOMBRE OBRERO MENOS PRODUCTIVO: XXXXXXXXXXXXXXXXXXXX
PRODUCCIÓ~~O~~N QUE FABRICÓ~~O~~: 999

Cálculos:
Se lee el nombre de un obrero, luego cada una de las cantidades producidas por día, se suman estas cantidades para calcular el TOTAL PRODUCCIÓ~~O~~N; en otras palabras, es la sumatoria de las cantidades producidas de todos los días que laboró.

El sueldo se calcula a 20.00 cada unidad fabricada si hizo 500 o menos; a 25.00 si hizo más de 500 y hasta 800; y a 30.00 si hizo más de 800.

Con formato: Sangría: Primera línea: 1.25 cm

Al final se pide el TOTAL de obreros, el TOTAL del TOTAL PRODUCCIÓ~~O~~N y el total de los sueldos de todos los obreros. Además el nombre del obrero más productivo y la producción que fabricó, y el nombre del obrero menos productivo y la producción que fabricó. El obrero más productivo es el que tenga el TOTAL PRODUCCIÓ~~O~~N mayor. El obrero menos productivo es el que tenga el TOTAL PRODUCCIÓ~~O~~N menor. Se supone que el TOTAL PRODUCCIÓ~~O~~N de los obreros e~~s~~s~~on~~ diferente~~s~~.

(Primero hágalo usted~~...~~, después compare la solución.)

```
Algoritmo OBREROS
1. Declarar
     Variables
       nombreObr, obrMayor, obrMenor: Cadena
       proDia, totProd, toTotProd, totObreros,
       mayorProd, menorProd: Entero
       sueldo, totSueldos: Real
       otro, desea: Carácter
2. Imprimir encabezado
3. totObreros = 0
   toTotProd = 0
   totSueldos = 0
   mayorProd = 0
   menorProd = 10000
4. do
     a. Solicitar Nombre
```

```
      b. Leer nombreObr
      c. totProd = 0
      d. do
           1. Solicitar Producción del diadía
           2. Leer proDia
           3. totProd = totProd + proDia
           4. Preguntar "¿Desea procesar otro día (S/N)?"
           5. Leer otro
      e. while otro == 'S'
      f. if totProd <= 500 then
            1. sueldo = totProd * 20.00
      g. endif
      h. if (totProd > 500) AND (totProd <= 800) then
            1. sueldo = totProd * 25.00
      i. endif
      j. if totProd > 800 then
            1. sueldo = totProd * 30.00
      k. endif
      l. Imprimir nombreObr, totProd, sueldo
      m. if totProd > mayorProd then
            1. mayorProd = totProd
            2. obrMayor = nombreObr
      n. endif
      o. if totProd < menorProd then
            1. menorProd = totProd
            2. obrMenor = nombreObr
      p. endif
      q. totObreros = totObreros + 1
         toTotProd = toTotProd + totProd
         totSueldos = totSueldos + sueldo
      r. Preguntar "¿Desea procesar otro obrero (S/N)?"
      s. Leer desea
   5. while desea == 'S'
   6. Imprimir totObreros, toTotProd, totSueldos,
              obrMayor, mayorProd, obrMenor, menorProd
   7. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C508.C y
Programa en Java: Obreros2.java

Explicación:
1. Se declaran las variables
2. Imprime encabezado
3. Se inician totObreros, toTotProd, totSueldos, mayorProd en ceros; y, menorProd en 10000
4. Inicia el ciclo do

a. Solicita el nombre
b. Lee en nombreObr
c. Inicia totProd en 0
d. Inicia el ciclo do
    1. Solicita la Producción del ~~dia~~día
    2. Lee en proDia
    3. Incrementa totProd con proDia
    4. Pregunta "¿Desea procesar otro d~~í~~a (S/N)?"
    5. Lee en otro la respuesta
e. Fin del ciclo while otro == 'S'~~,~~ ~~;~~ ~~S~~si se cumple, vuelve al do~~;~~ si no, sale del~~-~~ ciclo~~.~~
f. if totProd <= 500 then
    1. sueldo = totProd * 20.00
g. endif
h. if (totProd > 500) AND (totProd <= 800) then
    1. sueldo = totProd * 25.00
i. endif
j. if totProd > 800 then
    1. sueldo = totProd * 30.00
k. endif
l. Imprime nombreObr, totProd, sueldo
m. Compara si totProd > mayorProd entonces
    1. Coloca en mayorProd el totProd
    2. Coloca en obrMayor el nombreObr
n. Fin del if~~IF~~
o. Compara si totProd < menorProd entonces
    1. Coloca en menorProd el totProd
    2. Coloca en obrMenor el nombreObr
p. Fin del if~~IF~~
q. Se incrementa totObr en 1
   Se incrementa toTotProd en totProd
   Se incrementa totSueldos en sueldo
r. Pregunta "¿Desea procesar otro ~~obreo~~obrero (S/N)?"
s. Lee en desea la respuesta

5. Fin del ciclo while desea == 'S' ; Si se cumple, vuelve al do; si no, sale del ciclo.
6. Imprime totObreros, toTotProd, totSueldos,
        obrMayor, mayorProd, obrMenor, menorProd
7. Fin del algoritmo

Nota 1:
mayorProd se inicia en 0 antes del ciclo, para que cuando entre la primera vez al ciclo y procese la producción del primer obrero totProd y se compare en el if

```
m. if totProd > mayorProd then
      1. mayorProd = totProd
      2. obrMayor = nombreObr
n. endif
```

Con formato: Español (alfab. internacional)

Con formato: Diseño: Claro (Gris 12,5%)

Con formato: Justificado, Diseño: Claro (Gris 12,5%)

Con formato: Diseño: Claro (Gris 12,5%)

se cumpla la condición y entre al if a colocar los datos del primer obrero como el más productivo; y así, cuando dée la vuelta al ciclo y procese el segundo obrero, comparará el total de producción de este segundo obrero, con el del primero, que es el que está como el más productivo, si se cumple la condición entrará al if a colocar los datos del segundo obrero como el mayor, y si no se cumple la condición dejará el obrero que está como el mayor, luego hará lo propio con el siguiente obrero y así hasta que se terminen los obreros.

Nota 2:
menorProd se inicia en 10000 antes del ciclo, para que cuando entre la primera vez al ciclo y procese la producción del primer obrero totProd y se compare en el if

```
o. if totProd < menorProd then
      1. menorProd = totProd
      2. obrMenor = nombreObr
p. endif
```

se cumpla la condición y entre al if a colocar los datos del primer obrero como el menos productivo; y así, cuando dée la vuelta al ciclo y procese el segundo obrero, comparará el total de producción de este segundo obrero, con el del primero, que es el que está como el menos productivo, si se cumple la condición entrará al if a colocar los datos del segundo obrero como el menor, y si no se cumple la condición dejará el obrero que está como el menor, luego hará lo propio con el siguiente obrero y así hasta que se terminen los obreros. Es decir, 10000 es una cantidad mucho muy alta, que estamos seguros que ningún obrero lograría hacerla; de esta manera se fuerza a que la primera vez que llega al if, se cumpla la condición.

Ejercicio 5.1.2.7

En una empresa comercial, se tienen las compras de varios clientes, y por cada cliente los datos:

Nombre del cliente: X----------X
Cada cliente puede haber comprado varios artículos, por cada artículo que compró, se tienen los datos:

Artículo: X----------X
Cantidad de artículos comprados: ---
Precio unitario: ---

Nombre del cliente: X----------X
Artículo: X----------X
Cantidad de artículos comprados: ---
Precio unitario: ---

Con formato: Justificado, Diseño: Claro (Gris 12,5%)

Con formato: Diseño: Claro (Gris 12,5%)

Con formato: Justificado, Diseño: Claro (Gris 12,5%)

Con formato: Diseño: Claro (Gris 12,5%)

Con formato: Justificado, Diseño: Claro (Gris 12,5%)

Elaborar un algoritmo que lea los datos de cada uno de los clientes e imprima el siguiente reporte:

REPORTE DE VENTAS

NOMBRE	ARTÍCULO	TOTAL A PAGAR
XXXXXXXXXXXXXXXXXX	XXXXXXXXXXXXXXXXX	99,999.99
	XXXXXXXXXXXXXXXXX	99,999.99
	.	
	.	
	XXXXXXXXXXXXXXXXX	99,999.99
TOTAL CLIENTE		999,999.99
XXXXXXXXXXXXXXXXXX	XXXXXXXXXXXXXXXXX	99,999.99
	XXXXXXXXXXXXXXXXX	99,999.99
	.	
	.	
	XXXXXXXXXXXXXXXXX	99,999.99
TOTAL CLIENTE		999,999.99
.		
.		
.		
TOTAL CLIENTE		999,999.99
TOTAL GENERAL 999 CLIENTES		999,999.99

Cálculos:

TOTAL A PAGAR por cada artículo se calcula multiplicando la cantidad de artículos por el precio unitario.
TOTAL A PAGAR POR CLIENTE es la sumatoria del TOTAL A PAGAR de todos los artículos que compró dicho cliente.
En TOTAL GENERAL se imprime el total de clientes y el TOTAL A PAGAR, el cual es la sumatoria del TOTAL A PAGAR de todos los clientes.

(Primero hágalo usted, ...después compare la solución.)

```
Algoritmo CLIENTES CON ARTÍCULOS
1. Declarar
     Variables
       nombreClie: Cadena
       articulo: Cadena
       otro, hay: Carácter
       totClientes, cantidad: Entero
```

```
       precio, totPagar, totPagarClie, totPagarGral: Real
2. Imprimir encabezado
3. totClientes = 0
   totPagarGral = 0
4. do
     a. Solicitar Nombre del cliente
     b. Leer nombreClie
     c. Imprimir nombreClie
     d. totPagarClie = 0
     e. do
          1. Solicitar ArticuloArtículo, Cantidad, Precio
          2. Leer articuloartículo, cantidad, precio
          3. totPagar = cantidad * precio
          4. Imprimir artíiculo, totPagar
          5. totPagarClie = totPagarClie + totPagar
          6. Preguntar "¿Hay otro artíiculo(S/N)?"
          7. Leer hay
     f. while hay == 'S'
     g. Imprimir totPagarClie
     h. totClientes = totClientes + 1
        totPagarGral = totPagarGral + totPagarGralClie
     i. Preguntar "¿Hay otro cliente(S/N)?"
     j. Leer otro
5. while otro == 'S'
6. Imprimir totClientes, totPagarGral
7. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C509.C y
Programa en Java: Clientes2.java

Explicación:

1. Se declaran las variables
2. Imprime encabezado
3. Inicia totClientes y totPagarGral en ceros
4. Inicia ciclo do
   a. Se solicita Nombre del cliente
   b. Lee en nombreClie
   c. Imprime nombreClie
   d. Inicia totPagarClie en 0
   e. Inicia el ciclo do
      1. Solicita los datos ArticuloArtículo, Cantidad, Precio
      2. Lee en articuloartículo, cantidad, precio
      3. Calcula totPagar
      4. Imprime artíiculo, totPagar
      5. Incrementa totPagarClie en totPagar
      6. Pregunta "¿Hay otro artíiculo (S/N)?"

7. Lee en hay
f. Fin ciclo (while hay == 'S'), Si se cumple, vuelve al do, si no, sale del ciclo.
g. Imprime totPagarClie
h. Incrementa totClientes en 1
Incrementa totPagarGral en totPagarGralClie
i. Pregunta "¿Hay otro cliente (S/N)?"
j. Lee en otro la respuesta
5. Fin ciclo (while otro == 'S'), Si se cumple, vuelve al do, si no, sale del ciclo.
6. Imprime totClientes, totPagarGral
7. Fin del algoritmo

Ejercicio 5.1.2.8

En una compañía comercial, se tienen varios vendedores, y por cada vendedor los datos:

Nombre del vendedor: XXXXXXXXXXXXXXXXXXXXXXXXXX

Cada vendedor puede haber vendido varios artículos, por cada artículo que vendió, se tienen los datos:

Artículo: XXXXXXXXXXXXXXXXXXXXXXXXXX
Cantidad de artículos vendidos: 999
Precio unitario: 9,999.99
Clave de pago (1--4): 9

Nombre del vendedor: XXXXXXXXXXXXXXXXXXXXXXXXXX

Artículo: XXXXXXXXXXXXXXXXXXXXXXXXXX
Cantidad de artículos vendidos: 999
Precio unitario: 9,999.99
Clave de pago (1 -- 4): 9

Elaborar un algoritmo que lea los datos de cada uno de los vendedores e imprima el siguiente reporte:

REPORTE DE -INCENTIVOS

NOMBRE	TOTAL VENDIDO	INCENTIVO
XXXXXXXXXXXXXXXXXXX	99,999.99	99,999.99
XXXXXXXXXXXXXXXXXXX	99,999.99	99,999.99
.		
.		
.		
XXXXXXXXXXXXXXXXXXX	99,999.99	99,999.99
TOTAL 999 VENDEDORES	999,999.99	999,999.99

Cálculos:
TOTAL VENDIDO por cada vendedor es la sumatoria de lo vendido por el vendedor. Por cada artículo se calcula el importe vendido multiplicando la cantidad por precio unitario, y se suman los importes de todos los artículos vendidos.
INCENTIVO cada vez que se calcula el importe vendido de un artículo, es necesario calcular el incentivo que se le pagará por la venta de dicho artículo: si la clave de pago es 1, el incentivo es 15%, si la clave es 2, el incentivo es 10%, si es 3, el incentivo es 5% y si es 4, el incentivo es 3%.
TOTAL al final se imprime el total de vendedores, el total vendido por todos los vendedores y el total de incentivos de todos los vendedores.

(Primero hágalo usted, ...después compare la solución.)

```
Algoritmo VENDEDORES CON ARTÍCULOS
1. Declarar
     Variables
       nombreVend: Cadena
       articulo: Cadena
       otro, hay: Carácter
       totVend, cantidad, clave: Entero
       precio, totVendido, incentivo, totIncentivo,
       venta, totGralIncentivo, totGralVenta: Real
2. Imprimir encabezado
3. totVend = 0; totGralVenta = 0; totGralIncentivo = 0
4. do
     a. Solicitar Nombre del vendedor
     b. Leer nombreVend
     c. totVendido = 0; totIncentivo = 0
     d. do
          1. Solicitar ARTICULOARTÍCULO, CLAVE, CANTIDAD,
PRECIO
          2. Leer articuloartículo, clave, cantidad, precio
          3. venta = cantidad * precio
          4. switch clave
               1: incentivo = venta * 0.15
```

```
                2: incentivo = venta * 0.1
                3: incentivo = venta * 0.05
                4: incentivo = venta * 0.03
            5. endswitch
            6. totVendido = totVendido + venta
               totIncentivo = totIncentivo + incentivo
            7. Preguntar "¿Hay otro artíículo(S/N)?"
            8. Leer hay
        e. while hay == 'S'
        f. Imprimir nombreVend, totVendido, totIncentivo
        g. totVend = totVend + 1
           totGralVenta = totGralVenta + totVendido
           totGralIncentivo = totGralIncentivo +
                                          totIncentivo
        h. Preguntar "¿Hay otro vendedor(S/N)?"
        i. Leer otro
5. while otro == 'S'
6. Imprimir totVend, totGralVenta, totGralIncentivo
7. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C510.C y
Programa en Java: Vendedores1.java

Explicación:

1. Se declaran las variables
2. Imprime el encabezado
3. Se inician en cero totVend, totGralVenta, totGralIncentivo
4. Inicia ciclo do
   a. Solicita el Nombre del vendedor
   b. Se lee en nombreVend
   c. Se inician totVendido = 0; totIncentivo = 0
   d. Inicia ciclo do
      1. Se solicita ~~Articulo~~Artículo, Clave, Cantidad, Precio
      2. Se leen en ~~articulo~~artículo, clave, cantidad, precio
      3. Calcula venta = cantidad * precio
      4. Inicia switch clave
            1: Calcula incentivo = venta * 0.15
            2: Calcula incentivo = venta * 0.1
            3: Calcula incentivo = venta * 0.05
            4: Calcula incentivo = venta * 0.03
      5. Fin del switch
      6. Se incrementa totVendido con venta
         Se incrementa totIncentivo con incentivo
      7. Pregunta "¿Hay otro art~~i~~ículo (S/N)?"
      8. Lee en hay la respuesta
   e. Fin del ciclo (while hay == 'S')~~;~~, ~~S~~si se cumple, vuelve al do~~;~~, si no, sale del ciclo~~.~~

f. Imprime nombreVend, totVendido, totIncentivo
g. Incrementa totVend en 1
Incrementa totGralVenta con totVendido
Incrementa totGralIncentivo con totIncentivo
h. Pregunta "¿Hay otro vendedor (S/N)?"
i. Se lee en otro la respuesta
5. Fin del ciclo (while otro == 'S') ~~;~~ ~~S~~si se cumple, vuelve al do~~;~~ si no, sale del ciclo~~.~~
6. Imprime totVend, totGralVenta, totGralIncentivo
7. Fin del algoritmo

Nota:
Los problemas que hemos planteado para la aplicación de la repetición do…while, son situaciones en que algo se repite varias veces, no se sabe cuántas. Es por ello que se plantea la pregunta "¿Hay otro vendedor (S/N)?" y luego se lee la respuesta en otro, desea o hay.

Con formato: Diseño: Claro (Gris 12,5%)

Hay otro tipo de situaciones en que algo se repite un número conocido de veces; por ejemplo, tomando el ejercicio anterior ahora hagamos el mismo algoritmo pero considerando que se tienen 15 vendedores. ¿Cómo podríamos controlar la repetición del ciclo mediante una forma distinta?

Con formato: Sangría: Primera línea: 1.25 cm

~~.....~~ ¡Piénselo ~~.....~~ !

Si observamos el algoritmo, podemos ver que se tiene el contador `totVend` en el cual se cuenta la cantidad de vendedores que se están procesando; utilizando ~~é~~ese contador en el `while` quedaría de la forma siguiente:

```
Algoritmo VENDEDORES CON ARTÍ~~I~~CULOS
1. Declarar
     Variables
       nombreVend: Cadena
       articulo: Cadena
       hay: Carácter
       totVend, cantidad, clave: Entero
       precio, totVendido, incentivo, totIncentivo,
       venta, totGralIncentivo, totGralVenta: Real
2. Imprimir encabezado
3. totVend = 0; totGralVenta = 0; totGralIncentivo = 0
4. do
     a. Solicitar Nombre del vendedor
     b. Leer nombreVend
     c. totVendido = 0; totIncentivo = 0
     d. do
          1. Solicitar ~~Articulo~~Artículo, Clave, Cantidad,
Precio
```

```
        2. Leer articuloartículo, clave, cantidad, precio
        3. venta = cantidad * precio
        4. switch clave
             1: incentivo = venta * 0.15
             2: incentivo = venta * 0.1
             3: incentivo = venta * 0.05
             4: incentivo = venta * 0.03
        5. endswitch
        6. totVendido = totVendido + venta
           totIncentivo = totIncentivo + incentivo
        7. Preguntar "¿Hay otro artíículo(S/N)?"
        8. Leer hay
    e. while hay == 'S'
    f. Imprimir nombreVend, totVendido, totIncentivo
    g. totVend = totVend + 1
       totGralVenta = totGralVenta + totVendido
       totGralIncentivo = totGralIncentivo + totIncentivo
5. while totVend < 15
6. Imprimir totVend, totGralVenta, totGralIncentivo
7. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C511.C y
Programa en Java: Vendedores2.java

Observe que las acciones (del anterior algoritmo)

```
        h. Preguntar "¿HAY OTRO VENDEDOR(S/N)?"
        i. Leer otro
    5. while otro == 'S'
```

han sido cambiadas por (en este algoritmo):

```
    5. while totVend < 15
```

Simplemente se pregunta si totVend es menor a 15; si se cumple, significa que no se han procesado 15 vendedores, y regresa al do a procesar el siguiente. Si no se cumple,; se sale del ciclo, lo que quiere decir que ya se han procesado los 15 vendedores,

Con formato: Justificado

5.2.2 Ejercicios resueltos para la repetición for (Continuación...)

Ejercicio 5.2.2.9

Elaborar un algoritmo que pregunte a cuántos números se desea calcular el factorial; lea la cantidad en N. A continuación, debe leer un número e imprimir su factorial, luego leer otro, y así hasta leer los N números.

(Primero hágalo usted, ...después compare la solución.)

```
Algoritmo FACTORIALES DE N NÚUMEROS
1. Declarar
     Variables
       n, i, j, fact, num: Entero
2. Solicitar Cantidad de números a calcular factorial
3. Leer n
4. for j=1; j<=n; j++
     a. Solicitar Número
     b. Leer num
     c. if num == 0 then
           1. fact = 1
     d. else
           1. fact = 1
           2. for i=num; i>=1; i--
                 a. fact = fact * i
           3. endfor
     e. endif
     f. Imprimir fact
5. endfor
6. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C521.C y
Programa en Java: Factoriales1.java

Explicación:
1. Se declaran las variables
2. Se pregunta (solicita) la cantidad de números a calcular factorial
3. Se lee en n
4. Ciclo for desde j = 1 hasta n con incrementos de 1
   a. Se solicita el número
   b. Se lee en num
   c. Se compara si num == 0, si se cumple, entonces
      1. Se hace fact = 1
   d. Si no (else)
      1. Se inicia fact = 1
      2. Ciclo for desde i = num hasta 1 con decrementos de --1
         a. Se calcula fact = fact * i
      3. Fin del for
   e. Fin del if
   f. Se imprime fact

4. Fin del for
6. Fin del algoritmo

Ejercicio 5.2.2.10

Los números de Fibonacci constituyen una secuencia que empieza con 0 y 1; el número que sigue a éstos se calcula sumando los dos anteriores y así sucesivamente.
Elaborar un algoritmo que imprima los 20 siguientes números de la secuencia.

(Primero hágalo usted..., después compare la solución.)

```
Algoritmo SECUENCIA FIBONACCI
1. Declarar
     Variables
       i, numero, penultimo, ultimo: Entero
2. penultimopenúltimo = 0
   últimoltimo = 1
3. for i=1; i<=20; i++
       a. número = penultimopenúltimo + últimoltimo
       b. Imprimir número
       c. penultimopenúltimo = últimoltimo
       d. últimoltimo = número
4. endfor
5. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C522.C y
Programa en Java: Fibonacci.java

Expliciación:
1. Se declaran las variables
2. Se inicia el penultimopenúltimo en 0
   Se inicia el ultimoúltimo en 1
3. Ciclo for desde i = 1 hasta 20 con incrementos de 1
   a. Se calcula el nuevo número = penultimopenúltimo + últimoltimo
   b. Se imprime el nuevo numeronúmero
   c. En penultimopenúltimo se coloca el últimoltimo
   d. En ultimoúltimo se coloca el numeronúmero
4. Fin del for
5. Fin del algoritmo

Ejercicio 5.2.2.11

Elaborar un algoritmo que imprima una tabla con las potencias de los números desde 1 hasta 8. La potencia de 1, es 1 elevado a la potencia 1., La potencia de 2,

es 2 elevado a la potencia 2., Yy así sucesivamente, hasta la potencia de 8, es 8 elevado a la potencia 8.

Número	Potencia
1	9999
2	9999
-	
-	
8	9999

(Primero hágalo usted, ...después compare la solución.)

```
Algoritmo POTENCIAS 1-8
1. Declarar
     Variables
       numero, potencia: Real
         i: Entero
2. Imprimir encabezado
3. for numero=1; numero<=8; numero++
       a. potencia = número
       b. for i=1; i<numero; i++
              1. potencia = potencia * numero
       c. endfor
       d. Imprimir número, potencia
4. endfor
5. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C523.C y
Programa en Java: Potencias2.java

Explicación:
1. Se declaran las variables
2. Se imprime el encabezado
3. Inicia ciclo for desde número = 1 hasta 8 con incrementos de 1
   a. Se inicia potencia con numeronúmero
   b. Inicia ciclo for desde i = 1 hasta número --1 con incrementos de 1
      1. Calcula potencia = potencia * número
   c. Fin del for
   d. Imprime número, potencia
4. Fin del for
5. Fin del algoritmo

Ejercicio 5.2.2.12

Teniendo como datos de entrada valores para los coeficientes a, b, c, y aplicando la ecuación cuadrática: $F(x) = ax^2 + bx + c$.
Las raíces se calculan con la fórmula:

$$X = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Si $b^2 - 4ac = 0$ tiene raíz única, que se calcula $\frac{-b}{2a}$

Si $b^2 - 4ac$ es menor que 0 (cero) tiene raíces complejas, que se calculan:

Parte Real ± Parte Imaginaria

En donde:

Parte Real $= \frac{-b}{2a}$

Parte Imaginaria $= \frac{\sqrt{b^2 - 4ac}}{2a}$

Por lo que:
Raíz compleja 1 = Parte real + Parte imaginaria
Raíz compleja 2 = Parte real – Parte imaginaria
Si $b^2 - 4ac$ es mayor que 0 (cero) tiene raíces reales, aplicando la ecuación completa se calculan:

Raíz real 1 = $\frac{-b + \sqrt{b^2 - 4ac}}{2a}$

Raíz real 2 = $\frac{-b - \sqrt{b^2 - 4ac}}{2a}$

Con formato: Español (alfab. internacional)
Con formato: Español (alfab. internacional)
Con formato: Español (alfab. internacional)
Con formato: Español (alfab. internacional)
Con formato: Español (alfab. internacional)
Con formato: Español (alfab. internacional)
Con formato: Fuente: (Predeterminado) Arial, 12 pto, Disminuido 12 pto
Con formato: Centrado
Con formato: Español (alfab. internacional)
Con formato: Izquierda
Código de campo cambiado
Con formato: Fuente: (Predeterminado) Arial, 12 pto, Español (México), Disminuido 12 pto
Con formato: Español (alfab. internacional)
Con formato: Español (alfab. internacional)
Con formato: Español (alfab. internacional)
Con formato: Centrado
Con formato: Fuente: Cursiva
Con formato: Fuente: (Predeterminado) Arial, 12 pto, Disminuido 12 pto
Con formato: Fuente: (Predeterminado) Arial, 12 pto, Disminuido 12 pto
Con formato: Español (alfab. internacional)
Con formato: Español (alfab. internacional)
Con formato ... [1]
Con formato ... [2]
Con formato ... [3]
Con formato ... [4]
Con formato ... [5]
Con formato ... [6]
Código de campo cambiado
Con formato ... [7]
Con formato ... [8]

2a

Elaborar un algoritmo que permita que los valores de los coeficientes a, b, c, se comporten así: A debe ir de 1 hasta 5. C debe ir de 5 a 0. B debe tomar cada vez la diferencia de A – C, y que imprima para cada juego de valores de a, b, c, si tiene raíz única, raíces complejas o raíces reales, en una tabla como se muestra:

A	B	C	TIENE	RAÍZ 1	RAÍZ 2
1	-4	5	RAÍZ ÚNICA	999.99	
2	-2	4	RAÍCES COMPLEJAS	999.99	± 999.99
-	-	-	- - - -	- -	-
5	4	1	RAÍCES REALES	999.99	999.99

(Primero hágalo usted, ...después compare la solución.)

```
Algoritmo ECUACIÓN CUADRÁTICA PARA A DE 1-5
1. Declarar
     Variables
       a, b, c, raizUnica, parteReal, parteImaginaria,
       raizReal1, raizReal2: Real
2. Imprimir encabezado
3. c = 6
4. for a=1; a<=5; a++
       a. c = c - a
       b. b = a - c
       c. Imprimir a, b, c
       d. if (Potencia(b,2)-4*a*c) == 0 then
             1. raizUnica = -b/(2*a)
             2. Imprimir "RAÍZ ÚNICA  ", raizUnica
       e. else
             1. if (Potencia(b,2)-4*a*c) < 0 then
                  a. parteReal = -b/(2*a)
                 b. parteImaginaria =
                     RaizCuad(Absoluto(Potencia(b,2)-
                                       4*a*c))/(2*a)
                 c. Imprimir "RAÍCES COMPLEJAS"
                 d. Imprimir  parteReal,"+",
                                 parteImaginaria,"i"
                 e. Imprimir parteReal,"-",
                                 parteImaginaria,"i"
              2. else
                  a. raizReal1 =
                     (-b+RaizCuad(Potencia(b,2)-4*a*c))/(2*a)
                  b. raizReal2 =
                     (-b-RaizCuad(Potencia(b,2)-4*a*c))/(2*a)
                  c. Imprimir "RAÍCES REALES"
                  d. Imprimir raizReal1, raizReal2
```

Con formato: Justificado

Con formato: Español (alfab. internacional)

Con formato: Español (alfab. internacional)

Con formato: Español (alfab. internacional)

Con formato: Español (alfab. internacional)

```
            3. endif
         f. endif
5. endfor
6. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C524.C y
Programa en Java: Cuadratica2.java

Explicación:
1. Se declaran las variables
2. Se imprime el encabezado
3. Se inicia c en 6
4. Inicia ciclo for desde a = 1 hasta 5 con incrementos de 1
   a. Se calcula el coeficiente c = c – a
   b. Se calcula el coeficiente b = a – c
   c. Se imprimen los coeficientes a, b, c
   d. Si (Potencia(b,2) – 4*a*c) == 0 entonces
      1. Calcula la raizUnica
      2. Imprime "RAÍZ ÚNICA" , raizUnica
   e. Si no
      1. Si (Potencia(b,2) – 4*a*c) < 0 entonces
         a. Calcula parteReal
         b. Calcula parteImaginaria
         c. Calcula raizCompleja1
         d. Calcula raizCompleja2
         e. Imprime "RAÍCES COMPLEJAS"
         f. Imprime raizCompleja1, raizCompleja2
      2. Si no
         a. Calcula raizReal1
         b. Calcula raizReal2
         c. Imprime "RAÍCES REALES"
         d. Imprime raizReal1, raizReal2
      3. Fin del if
   f. Fin del if
5. Fin del for
6. Fin del algoritmo

Ejercicio 5.2.2.13

Se deposita la cantidad de 10 000.00 a un plazo de 24 meses, con una tasa de interés de 36% anual, capitalizable cada mes, es decir, se suman los intereses ganados al capital. Elaborar un algoritmo que imprima el siguiente reporte:

INVERSIÓN

MES CAPITAL INTERÉS SALDO

1	99,999.99	99,999.99	99,999.99
2	99,999.99	99,999.99	99,999.99
.			
.			
.			
24	99,999.99	99,999.99	99,999.99

TOTAL INTERÉS GANADO: 99,999.99

Cálculos:
MES es el número de mes.
CAPITAL es el capital que se tiene en cada mes, por ejemplo, en el mes 1, el capital es 10,000.00, para el mes 2, deberá tener incrementado el interés que ganó en el mes 1, y así sucesivamente.
INTERÉS es el interés que gana en el mes correspondiente, aplicándole al capital del mes, la tasa de interés mensual que le corresponda (36/12).
SALDO es el saldo de la inversión del mes correspondiente; se suman el capital y el interés.

(Primero hágalo usted, ...después compare la solución.)

```
Algoritmo INVERSION 24 MESES
1. Declarar
     Variables
       mes: Entero
       saldo , capital, interes, totInteres: Real
2. Imprimir encabezado
3. capital = 10000.00
4. totInteres = 0
5. for mes=1; mes<=24; mes++
       a. interesinterés = capital * (0.36/12)
       b. saldo = capital + interesinterés
       c. Imprimir mes, capital, interesinterés, saldo
       d. totInteres = totInteres + interesinterés
       e. capital = saldo
6. endfor
7. Imprimir totInteres
8. Fin
```

Con formato: Fuente: 12 pto

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C525.C y
Programa en Java: Interes1.java

Explicación:
1. Se declaran las variables

2. Se imprime el encabezado
3. Se inicia capital en 10 000.00
4. Se inicia totInteres en 0
5. Ciclo for desde mes = 1 hasta 24 con incrementos de 1
   a. Se calcula el ~~interes~~interés del mes
   b. Se calcula el saldo del mes (capital + ~~interes~~interés)
   c. Se imprime mes, capital, ~~interes~~interés, saldo
   d. Se calcula el totInteres con ~~interes~~interés
   e. El capital se actualiza con saldo
6. Fin del for
7. Se Imprime totInteres
8. Fin del algoritmo

Ejercicio 5.2.2.14

Similar al anterior, pero ahora el capital, la tasa de interés anual y el plazo son datos que deben leerse. Elaborar un algoritmo que lea dichos datos e imprima el siguiente reporte:

.

INVERSI~~O~~ÓN
CAPITAL: 99,999.99
INTERÉ~~E~~S ANUAL: 999.99%
PLAZO EN MESES: 999

MES	CAPITAL	INTERÉ~~E~~S	SALDO
1	99,999.99	99,999.99	99,999.99
2	99,999.99	99,999.99	99,999.99
.			
.			
.			
N	99,999.99	99,999.99	99,999.99

TOTAL INTEREÉS GANADO: 99,999.99

(Primero hágalo usted, ~~...~~después compare la solución.)

```
Algoritmo INVERSIÓ~~O~~N LEYENDO DATOS
1. Declarar
     Variables
        mes, plazo: Entero
        saldo, capital, interes, totInteres, tasaAnual: Real
2. Solicitar Capital, Tasa interés anual, Plazo
3. Leer capital, tasaAnual, plazo
4. Imprimir capital, tasaAnual, plazo
5. Imprimir encabezado
6. totInteres = 0
```

```
7. for mes=1; mes<=plazo; mes++
       a. interesinterés = capital * (tasaAnual/100/12)
       b. saldo = capital + interesinterés
       c. Imprimir mes, capital, interesinterés, saldo
       d. totInteres = totInteres + interesinterés
       e. capital = saldo
8. endfor
9. Imprimir totInteres
10. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C526.C y
Programa en Java: Interes2.java

Explicación:

1. Se declaran las variables
2. Se Ssolicita Capital, Tasa interés anual, Plazo
3. Se leen en capital, tasaAnual, plazo
4. Se imprimen capital, tasaAnual, plazo
5. Imprime el encabezado
6. Se inicia totInteres en 0
7. Ciclo for desde mes = 1 hasta plazo, con incrementos de 1
   a. Se calcula el interesinterés del mes
   b. Se calcula el saldo (capital + interesinterés)
   c. Se imprimen mes, capital, interes, saldo
   d. Se calcula incrementa totInteres en interesinterés
   e. El capital se actualiza con saldo
8. Fin del for
9. Se imprime totInteres
10. Fin del algoritmo

Ejercicio 5.2.2.15

En una empresa manufacturera de sillas., Sse tienen 15 trabajadores, y por cada trabajador los datos:

Nombre del trabajador 1: X----------X

Se tiene el dato: cantidad de unidades fabricadas (construidas) en cada uno de los 6 días de la semana.

Día 1 (Cantidad de unidades fabricadas): ---
Día 2 (Cantidad de unidades fabricadas): ---
Día 3 (Cantidad de unidades fabricadas): ---
Día 4 (Cantidad de unidades fabricadas): ---
Día 5 (Cantidad de unidades fabricadas): ---
Día 6 (Cantidad de unidades fabricadas): ---

Nombre del trabajador 2: X----------X
Día 1 (Cantidad de unidades fabricadas): ---
Día 2 (Cantidad de unidades fabricadas): ---
Día 3 (Cantidad de unidades fabricadas): ---
Día 4 (Cantidad de unidades fabricadas): ---
Día 5 (Cantidad de unidades fabricadas): ---
Día 6 (Cantidad de unidades fabricadas): ---

Nombre del trabajador 15: X----------X
Día 1 (Cantidad de unidades fabricadas): ---
Día 2 (Cantidad de unidades fabricadas): ---
Día 3 (Cantidad de unidades fabricadas): ---
Día 4 (Cantidad de unidades fabricadas): ---
Día 5 (Cantidad de unidades fabricadas): ---
Día 6 (Cantidad de unidades fabricadas): ---

Elaborar un algoritmo que lea los datos de cada uno de los trabajadores, y por cada trabajador, la cantidad de unidades fabricadas de cada uno de los 6 días que trabajó e imprima el siguiente reporte:

REPORTE SEMANAL DE PRODUCCIÓN

NOMBRE DEL TRABAJADOR	TOTAL PRODUCCIÓN
XXXXXXXXXXXXXXXXXXXXXXX	9999
XXXXXXXXXXXXXXXXXXXXXXX	9999
.	
.	
XXXXXXXXXXXXXXXXXXXXXXX	9999
TOTAL 999 TRABAJADORES	9999

Cálculos:
Se lee el nombre de un trabajador, luego cada una de las cantidades producidas, se suman estas cantidades para calcular TOTAL ~~PRODUCCION ;~~PRODUCCION; en otras palabras, es la sumatoria de las cantidades producidas de los 6 días que laboró.
Al final se pide el TOTAL de trabajadores y el TOTAL del TOTAL PRODUCCIÓN de todos los trabajadores.

(Primero hágalo usted, ~~...~~después compare la solución.)

```
Algoritmo TRABAJADORES
1. Declarar
     Variables
       nombreTra: Cadena
       t, d, totTrab, proDia,
```

```
        totProd, totProdGral: Entero
2.  Imprimir encabezado
3.  totTrab = 0; toTotProd = 0
4.  for t=1; t<=15; t++
        a. Solicitar Nombre
        b. Leer nombreTra
        c. totProd =
        d. for d=1; d<=6; d++
               1. Solicitar PRODUCCIÓON DEL DÍA
               2. Leer proDia
               3. totProd = totProd + proDia
        e. endfor
        f. Imprimir nombreTra, totProd
        g. toTotProd = toTotProd + totProd
           totTrab = totTrab + 1
5.  endfor
6.  Imprimir totTrab, toTotProd
7.  Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C527.C y
Programa en Java: Trabajadores.java

Explicación:

1. Se declaran las variables
2. Se imprime el encabezado
3. Se inicia totTrab = 0; toTotProd = 0
4. Inicia ciclo for desde t = 1 hasta 15 con incrementos de 1
   a. Se solicita el Nombre
   b. Se lee en nombreTra
   c. Se inicia totProd = 0
   d. Inicia ciclo for desde d = 1 hasta 6 con incrementos de 1
      1. Se solicita Produccion del dia
      2. Se lee en proDia
      3. Se incrementa totProd con proDia
   e. Fin del for
   f. Imprime nombreTra, totProd
   g. Se incrementa toTotProd con totProd
      Se incrementa totTrab en 1
5. Fin del for
6. Imprime totTrab, toTotProd
7. Fin del algoritmo

Nota:
Observe que este problema se resuelve utilizando dos ciclos for; un primer ciclo que permite procesar varios obreros. Al procesar cada obrero, después de leer su nombre, se debe procesar varios días de trabajo, para lo cual, se utiliza un segundo ciclo for.

Con formato: Justificado, Diseño: Claro (Gris 12,5%)

Con formato: Diseño: Claro (Gris 12,5%)

Además observe que no todos los contadores y acumuladores se inician en cero en el mismo lugar; antes del primer ciclo se inician los contadores y acumuladores que se imprimirán hasta el final del reporte, como son totTrab y toTotProd; los contadores y acumuladores que se usarán o imprimirán en el detalle del reporte como es el totProd, se inicia en cero antes del segundo ciclo (pero adentro del primer ciclo), que se incrementa adentro del segundo ciclo con proDia y que se imprime afuera del segundo ciclo, pero adentro del primer ciclo.

Con formato: Sangría: Primera línea: 1.25 cm, Diseño: Claro (Gris 12,5%)

Ejercicio 5.2.2.16

Se tienen varios vendedores, por cada vendedor los datos: nombre y la cantidad vendida (en dinero) de cada uno de los 6 días de la semana. Elaborar un algoritmo que lea dichos datos e imprima el reporte:

```
                    REPORTE  SEMANAL  VENTAS
NOMBRE DEL VENDEDOR      VENTA  SEMANAL    DÍA DE MAYOR VENTA
-------------------------------------------------------------------------------------------------------
XXXXXXXXXXXXXXXXXXXX        999,999.99                99
XXXXXXXXXXXXXXXXXXXX        999,999.99                99
            .
            .
            .
XXXXXXXXXXXXXXXXXXXX        999,999.99                99

TOTAL  999  VENDEDORES          999,999.99
```

Se lee el nombre del vendedor, luego la venta que hizo en cada uno de los 6 días de la semana, se suman para calcular la VENTA SEMANAL, el DÍA DE MAYOR VENTA se determina comparando la venta de los 6 días entre sí, obteniendo el día que tuvo la mayor venta. Al final se pide el total de vendedores y el total general de venta semanal.

(Primero hágalo usted, ...después compare la solución.)

```
Algoritmo VENDEDORES MAYOR VENTA DÍA
1. Declarar
     Variables
       nombreVend: Cadena
       otro: Caráácter
       totVend, dia, mejorDia: Entero
       ventaDia, totVenta, totVentaGral, mayorVenta: Real
2. Imprimir encabezado
3. totVend = 0; totVentaGral = 0
4. do
     a. Solicitar Nombre
     b. Leer nombreVend
     c. totVenta = 0; mayorVenta = 0
```

```
      d. for dia=1; dia<=6; dia++
            1. Solicitar VENTA DEL DÍA
            2. Leer ventaDia
            3. totVenta = totVenta + ventaDia
            4. if ventaDia > mayorVenta then
                  a. mayorVenta = ventaDia
                  b. mejorDia = dia
            5. endif
      e. endfor
      f. Imprimir nombreVend, totVenta, mejorDia
      g. totVentaGral = totVentaGral + totVenta
         totVend = totVend + 1
      h. Preguntar "¿Hay otro vendedor (S/N)?"
      i. Leer otro
5. while otro == 'S'
6. Imprimir totVend, totVentaGral
7. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C528.C y
Programa en Java: Vendedores3.java

Explicación:

1. Se declaran las variables
2. Se imprime el encabezado
3. Se inician totVend = 0; totVentaGral = 0
4. Inicia ciclo do
   a. Solicita el Nombre
   b. Se lee en nombreVend
   c. Se inicia totVenta = 0; mayorVenta = 0
   d. Inicia for desde ~~dia~~día = 1 hasta 6 con incrementos de 1
      1. Se solicita Venta del ~~dia~~día
      2. Se lee ventaDia
      3. Se incrementa totVenta con ventaDia
      4. Si acaso ventaDia > mayorVenta entonces
         a. Coloca en mayorVenta la ventaDia
         b. Coloca en mejorDia el ~~dia~~día
      5. Fin del if
   e. Fin del for
   f. Imprime nombreVend, totVenta, mejorDia
   g. Incrementa totVentaGral con totVenta
      Incrementa totVend en 1
   h. Pregunta "¿Hay otro vendedor (S/N)?"
   i. Se lee en otro la respuesta
5. Fin ciclo (while otro == 'S')
6. Imprime totVend, totVentaGral
7. Fin del algoritmo

Nota:
Observe que este problema se resuelve con dos ciclos; un primer ciclo do…while, porque se procesarán varios vendedores; y un segundo ciclo for, porque dentro del proceso de un vendedor, se procesarán 6 días de venta.

Con formato: Justificado, Diseño: Claro (Gris 12,5%)

Ejercicio 5.2.2.17

Elaborar un algoritmo similar al Ejercicio 5.1.2.6, con la diferencia de que ahora se tienen 15 obreros y cada obrero trabajó 6 días.

(Primero hágalo usted, ~~...~~después compare la solución.)

```
Algoritmo OBREROS
1. Declarar
     Variables
       nombreObr, obrMayor, obrMenor: Cadena
       proDia, totProd, toTotProd, totObreros,
       i, d~~i~~ías, mayorProd, menorProd: Entero
       sueldo, totSueldos: Real
2. Imprimir encabezado
3. totObreros = 0
   toTotProd = 0
   totSueldos = 0
   mayorProd = 0
   menorProd = 10000
4. for i=1; i<=15; i++
       a. Solicitar Nombre
       b. Leer nombreObr
       c. totProd = 0
       d. for d=1; d<=6; d++
            1. Solicitar Producción del ~~dia~~día
            2. Leer proDia
            3. totProd = totProd + proDia
       e. endfor
       f. if totProd <= 500 then
             1. sueldo = totProd * 20.00
       g. endif
       h. if (totProd > 500) AND (totProd <= 800) then
             1. sueldo = totProd * 25.00
       i. endif
       j. if totProd > 800 then
             1. sueldo = totProd * 30.00
       k. endif
       l. Imprimir nombreObr, totProd, sueldo
       m. if totProd > mayorProd then
             1. mayorProd = totProd
```

```
            2. obrMayor = nombreObr
        n. endif
        o. if totProd < menorProd then
            1. menorProd = totProd
            2. obrMenor = nombreObr
        p. endif
        q. totObreros = totObreros + 1
           toTotProd = toTotProd + totProd
           totSueldos = totSueldos + sueldo
5. endfor
6. Imprimir totObreros, toTotProd, totSueldos,
           obrMayor, mayorProd, obrMenor, menorProd
6. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C529.C y
Programa en Java: Obreros3.java

Explicación:
Lo que cambia es el primer ciclo, que en el ejercicio 5.1.2.6 se había hecho con do…while usando la pregunta y leer desea, ahora se resuelve con for usando la variable i para controlar el proceso de los 15 obreros. Asimismo, el segundo ciclo en lugar de do…while usando la pregunta y leer otro, ahora se resuelve con for usando la variable d para controlar el proceso de los 6 días en que trabajó cada obrero.

Lo relevante de este algoritmo es que aquí se combina el uso de un for con un do…while adentro.

Con formato: Sangría: Primera línea: 1.25 cm

Ejercicio 5.2.2.18

Elaborar un algoritmo similar al Ejercicio 5.1.2.6, con la diferencia de que ahora se tienen 15 obreros y cada obrero trabajó varios días.

(Primero hágalo usted, ...después compare la solución.)

```
Algoritmo OBREROS
1. Declarar
     Variables
       nombreObr, obrMayor, obrMenor: Cadena
       proDia, totProd, toTotProd, totObreros,
       i, mayorProd, menorProd: Entero
       sueldo, totSueldos: Real
       otro: Caraácter
2. Imprimir encabezado
3. totObreros = 0
   toTotProd = 0
   totSueldos = 0
   mayorProd = 0
```

```
   menorProd = 10000
4. for i=1; i<=15; i++
       a. Solicitar Nombre
       b. Leer nombreObr
       c. totProd = 0
       d. do
            1. Solicitar Producción del diadía
            2. Leer proDia
            3. totProd = totProd + proDia
            4. Preguntar "¿Desea procesar otro díia (S/N)?"
            5. Leer otro
       e. while otro == 'S'
       f. if totProd <= 500 then
             1. sueldo = totProd * 20.00
       g. endif
       h. if (totProd > 500) AND (totProd <= 800) then
             1. sueldo = totProd * 25.00
       i. endif
       j. if totProd > 800 then
             1. sueldo = totProd * 30.00
       k. endif
       l. Imprimir nombreObr, totProd, sueldo
       m. if totProd > mayorProd then
             1. mayorProd = totProd
             2. obrMayor = nombreObr
       n. endif
       o. if totProd < menorProd then
             1. menorProd = totProd
             2. obrMenor = nombreObr
       p. endif
       q. totObreros = totObreros + 1
          toTotProd = toTotProd + totProd
          totSueldos = totSueldos + sueldo
5. endfor
6. Imprimir totObreros, toTotProd, totSueldos,
            obrMayor, mayorProd, obrMenor, menorProd
6. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C530.C y
Programa en Java: Obreros4.java

Explicación:
Lo que cambia es el primer ciclo, que en el ejercicio 5.1.2.6 se había hecho con do…while usando la pregunta y leer desea, ahora se resuelve con for usando la variable i para controlar el proceso de los 15 obreros.

Lo relevante de este algoritmo es que aquí se combina el uso de un for con un do…while adentro.

Con formato: Sangría: Primera línea: 1.25 cm

Ejercicio 5.2.2.19

Elaborar un algoritmo similar al Ejercicio 5.1.2.6, con la diferencia de que ahora se tienen varios obreros y cada obrero trabajó 6 días.

(Primero hágalo usted, ...después compare la solución.)

```
Algoritmo OBREROS
1. Declarar
     Variables
       nombreObr, obrMayor, obrMenor: Cadena
       proDia, totProd, toTotProd, totObreros,
       i, mayorProd, menorProd: Entero
       sueldo, totSueldos: Real
       desea: Caráacter
2. Imprimir encabezado
3. totObreros = 0
   toTotProd = 0
   totSueldos = 0
   mayorProd = 0
   menorProd = 10000
4. do
     a. Solicitar Nombre
     b. Leer nombreObr
     c. totProd = 0
     d. for i=1; i<=6; i++
          1. Solicitar Producción del diadía
          2. Leer proDia
          3. totProd = totProd + proDia
     e. endfor
     f. if totProd <= 500 then
           1. sueldo = totProd * 20.00
     g. endif
     h. if (totProd > 500) AND (totProd <= 800) then
           1. sueldo = totProd * 25.00
     i. endif
     j. if totProd > 800 then
           1. sueldo = totProd * 30.00
     k. endif
     l. Imprimir nombreObr, totProd, sueldo
     m. if totProd > mayorProd then
           1. mayorProd = totProd
           2. obrMayor = nombreObr
     n. endif
```

```
     o. if totProd < menorProd then
           1. menorProd = totProd
           2. obrMenor = nombreObr
     p. endif
     q. totObreros = totObreros + 1
        toTotProd = toTotProd + totProd
        totSueldos = totSueldos + sueldo
     r. Preguntar "¿Desea procesar otro obrero (S/N)?"
     s. Leer desea
5. while desea == 'S'
6. Imprimir totObreros, toTotProd, totSueldos,
            obrMayor, mayorProd, obrMenor, menorProd
7. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C531.C y
Programa en Java: Obreros5.java

Explicación:
Lo que cambia es el segundo ciclo, que en el ejercicio 5.1.2.6 se había hecho con do…while usando la pregunta y leer otro, ahora se resuelve con for usando la variable i para controlar el proceso de los 6 días.

Lo relevante de este algoritmo es que aquí se combina el uso de un do…while con un for adentro.

Con formato: Sangría: Primera línea: 1.25 cm

Ejercicios del punto 5.2.3 Simulación del for con do…while (Continuación…)

Ejercicio 5.2.3.2

Elaborar un algoritmo similar al Ejercicio 5.2.2.6, utilizando do…while en lugar de for.

```
Algoritmo EQUIVALENCIAS FAHRENHEIT CELSIUS
1. Declarar
      Variables
        fahrenheit, celsius: Real
2. Imprimir encabezado
3. fahrenheit = 0
4. do
     a. fahrenheit = fahrenheit + 1
     b. celsius = (fahrenheit-32)*(5/9)
     c. Imprimir fahrenheit, celsius
5. while fahrenheit < 65
6. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C533.C y

Programa en Java: EquivalenciasFahr2.java

Ejercicio 5.2.3.3

Elaborar un algoritmo similar al Ejercicio 5.2.2.9, utilizando do…while en lugar de for.

```
Algoritmo FACTORIALES DE N NÚUMEROS
1. Declarar
     Variables
       n, i, j, fact, num: Entero
2. Solicitar Cantidad de números a calcular factorial
3. Leer n
4. j = 0
5. do
     a. j = j + 1
     b. Solicitar Número
     c. Leer num
     d. if num == 0 then
           1. fact = 1
     e. else
           1. fact = 1
           2. for i=num; i>=1; i--
                 a. fact = fact * i
           3. endfor
     f. endif
     g. Imprimir fact
6. while j < n
7. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C534.C y
Programa en Java: Factoriales2.java

Ejercicio 5.2.3.4

Elaborar un algoritmo similar al Ejercicio 5.2.2.12, utilizando do…while en lugar de for.

```
Algoritmo ECUACIOÓN CUADRAÁTICA PARA A DE 1-5
1. Declarar
     Variables
       a, b, c, raizUnica, parteReal, parteImaginaria,
       raizReal1, raizReal2: Real
2. Imprimir encabezado
3. c = 6
4. a = 0
```

```
5. do
     a. a = a + 1
     b. c = c - a
     c. b = a - c
     d. Imprimir a, b, c
     e. if (Potencia(b,2)-4*a*c) == 0 then
           1. raizUnica = -b/(2*a)
           2. Imprimir "RAÍZ ÚNICA  ", raizUnica
     f. else
           1. if (Potencia(b,2)-4*a*c) < 0 then
                a. parteReal = -b/(2*a)
                b. parteImaginaria =
                   RaizCuad(Absoluto(Potencia(b,2)-
                                     4*a*c))/(2*a)
                c. Imprimir "RAÍCES COMPLEJAS"
                d. Imprimir parteReal,"+",parteImaginaria,"i"
                e. Imprimir parteReal,"-",parteImaginaria,"i"
           2. else
                 a. raizReal1 =
                     (-b+RaizCuad(Potencia(b,2)-4*a*c))/(2*a)
                 b. raizReal2 =
                     (-b-RaizCuad(Potencia(b,2)-4*a*c))/(2*a)
                 c. Imprimir "RAICES REALES"
                 d. Imprimir raizReal1, raizReal2
           3. endif
     g. endif
6. while a < 5
7. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C535.C y
Programa en Java: Cuadratica3.java

5.3.3 Ejercicios resueltos para la repetición while (Continuación...)

Con formato: Fuente: Sin Negrita

Ejercicio 5.3.3.3

Una compañía manufacturera fabrica el producto A. Para fabricar una unidad de dicho producto se requieren los siguientes materiales:

Material 1: 3 unidades
Material 2: 4 unidades
Material 3: 1 unidades
Material 4: 2 unidades
Material 5: 3 unidades
Material 6: 2 unidades

Elaborar un algoritmo que lea pedidos del producto A, en cada pedido se tiene el dato cantidad de unidades del producto A; cuando termine de leer los pedidos, imprimir:

LISTADO DE MATERIALES REQUERIDOS

MATERIAL	TOTAL DE UNIDADES
1	999
2	999
3	999
4	999
5	999
6	999

(Primero hágalo usted, ...después compare la solución).

```
Algoritmo MATERIAL REQUERIDO
1. Declarar
     Variables
       resp: Caracter
       material1, material2, material3, material4,
       material5, material6, cantidad, totProd: Entero
2. totProd = 0
3. Preguntar "¿Hay pedido (S/N)?"
4. Leer resp
5. while resp == 'S'
     a. Solicitar Cantidad pedida
     b. Leer cantidad
     c. totProd = totProd + cantidad
     d. Preguntar "¿Hay otro pedido (S/N)?"
     e. Leer resp
6. endwhile
7. material1 = totProd * 3
   material2 = totProd * 4
   material3 = totProd * 1
   material4 = totProd * 2
   material5 = totProd * 3
   material6 = totProd * 2
8. Imprimir encabezado
9. Imprimir 1, material1,
            2, material2,
            3, material3,
            4, material4,
            5, material5,
            6, material6
10. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C542.C y
Programa en Java: Materiales.java

Explicación:
1. Se declaran las variables
2. Inicia totProd en 0
3. Pregunta "¿Hay pedido (S/N)?"
4. Lee en resp la respuesta
5. Inicia ciclo while mientras resp es igual a 'S'; entra al ciclo
   a. Solicita cantidad pedida
   b. Lee en cantidad
   c. Incrementa totProd en cantidad
   d. Pregunta "¿Hay otro pedido (S/N)?"
   e. Lee en resp la respuesta
6. Fin del ciclo while
7. Calcula material1 = totProd * 3
   material2 = totProd * 4
   material3 = totProd * 1
   material4 = totProd * 2
   material5 = totProd * 3
   material6 = totProd * 2
8. Imprime encabezado
9. Imprime 1, material1,
   2, material2,
   3, material3,
   4, material4,
   5, material5,
   6, material6
10. Fin del algoritmo

Nota:
El ciclo que procesa los pedidos, sólo puede resolverse con while porque la repetición que se presenta puede ir desde 0 (cero) hasta cualquier cantidad de pedidos.

Con formato: Diseño: Claro (Gris 12,5%)

Ejercicio 5.3.3.4

En una compañía manufacturera se tienen 10 máquinas, y hay 7 tipos de mantenimiento correctivo que se les aplican. Se tienen los datos:

Máquina 1
Mantenimiento tipo: --
Mantenimiento tipo: --
.
Mantenimiento tipo: --

Máquina 2
   Mantenimiento tipo: --
   Mantenimiento tipo: --
        .
   Mantenimiento tipo: --

Máquina 10
   Mantenimiento tipo: --
   Mantenimiento tipo: --
        .
   Mantenimiento tipo: --

Para cada máquina se pueden tener cualquier cantidad (incluso cero) de mantenimientos, teniendose como dato la identificación del tipo de mantenimiento (1,2,3,4,5,6,7). Elaborar un algoritmo que lea dichos e imprima el reporte:

REPORTE DE MANTENIMIENTO

MÁAQUINA	TOTAL DE MANTENIMIENTOS
1	9999
2	9999
.	
.	
10	9999
TOTAL MANTENIMIENTOS	9999

TOTAL TIPO 1: ---
TOTAL TIPO 2: ---
TOTAL TIPO 3: ---
TOTAL TIPO 4: ---
TOTAL TIPO 5: ---
TOTAL TIPO 6: ---
TOTAL TIPO 7: ---

(Primero hágalo usted, ...después compare la solución) .)

Con formato: Fuente: Cursiva

```
Algoritmo MANTENIMIENTO MAÁQUINAS
1. Declarar
     Variables
       hay: Caraácter
       totManten, cantManten, tipo, maquina, toTipo1,
       toTipo2, toTipo3, toTipo4, toTipo5, toTipo6,
       toTipo7: Entero
```

```
2. Imprimir encabezado
3. totManten = 0; toTipo1 = 0; toTipo2 = 0; toTipo3=0;
   toTipo4 = 0; toTipo5 = 0; toTipo6 = 0; toTipo7 = 0;
4. for maquina=1; maquina<=10; maquina++
       a. cantManten = 0
       b. Preguntar "¿Hay mantenimiento (S/N)?"
       c. Leer hay
       d. while hay == 'S'
            1. Solicitar Tipo de mantenimiento
            2. Leer tipo
            3. cantManten = cantManten + 1
            4. switch tipo
                  1: toTipo1 = toTipo1 + 1
                  2: toTipo2 = toTipo2 + 1
                  3: toTipo3 = toTipo3 + 1
                  4: toTipo4 = toTipo4 + 1
                  5: toTipo5 = toTipo5 + 1
                  6: toTipo6 = toTipo6 + 1
                  7: toTipo7 = toTipo7 + 1
            5. endswitch
            6. Preguntar "¿Hay otro mantenimiento(S/N)?"
            7. Leer hay
       e. endwhile
       f. Imprimir máaquina, cantManten
       g. totManten = totManten + cantManten
5. endfor
6. Imprimir totManten, toTipo1, toTipo2, toTipo3,
           toTipo4, toTipo5, toTipo6, toTipo7
7. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C543.C y
Programa en Java: Maquinas.java

Explicación:

1. Se declaran las variables
2. Imprime encabezado
3. Inicia totManten en 0; toTipo1 en 0; toTipo2 en 0; toTipo3 en 0;
   toTipo4 en 0; toTipo5 en 0; toTipo6 en 0; toTipo7 en 0;
4. Inicia ciclo for desde máaquina = 1 hasta 10 con incrementos de 1
   a. Inicia cantManten en 0
   b. Pregunta "¿Hay mantenimiento (S/N)?"
   c. Lee en hay la respuesta
   d. Inicia ciclo while mientras hay == 'S'; entra al ciclo
      1. Solicita tipo de mantenimiento
      2. Lee en tipo
      3. Incrementa cantManten en 1

4. Casos (switch)
   Si tipo es 1: Incrementa toTipo1 en 1
   Si tipo es 2: Incrementa toTipo2 en 1
   Si tipo es 3: Incrementa toTipo3 en 1
   Si tipo es 4: Incrementa toTipo4 en 1
   Si tipo es 5: Incrementa toTipo5 en 1
   Si tipo es 6: Incrementa toTipo6 en 1
   Si tipo es 7: Incrementa toTipo7 en 1
5. Fin del switch
6. Pregunta "¿Hay otro mantenimiento (S/N)?"
7. Lee en hay la respuesta

e. Fin del ciclo while
f. Imprime máquina, mantManten
g. Incrementa totManten con cantManten

5. Fin ciclo for
6. Imprime totManten, toTipo1, toTipo2, toTipo3,
   toTipo4, toTipo5, toTipo6, toTipo7
7. Fin del algoritmo

Nota:
El ciclo que procesa los mantenimientos, sólo puede resolverse con while porque la repetición que se presenta puede ir desde 0 (cero) hasta cualquier cantidad de mantenimientos.

Con formato: Diseño: Claro (Gris 12,5%)

Ejercicio 5.3.3.5

En una fábrica se tienen 10 estaciones de trabajo, y la producción hecha en número de unidades fabricadas en cada uno de los días de la semana; cada estación pudo haber trabajado cualquier cantidad de días en la semana.

Estación 1:
   Producción ~~dia~~día: ---
   Producción ~~dia~~día: ---
      .
      .
   Producción ~~dia~~día: ---

Estación 2:
   Producción ~~dia~~día: ---
   Producción ~~dia~~día: ---
      .
      .
   Producción ~~dia~~día: ---

Estación 10:
   Producción ~~dia~~día: ---
   Producción ~~dia~~día: ---
      .

.

Producción diadía: ---

También se tiene un dato que es el nivel de productividad.

Elaborar un algoritmo que lea el nivel de productividad, y los datos de la producción de cada una de las estaciones, e imprima el siguiente reporte:

REPORTE DE PRODUCCIOÓN

ESTACIÓON	TOTAL PRODUCCIÓON	NIVEL PRODUCTIVIDAD
1	999	DEFICIENTE
2	999	BUENO
3	-----	
.	-----	
.	-----	
10	999	EXCELENTE
TOTAL	999	

TOTAL PRODUCCIÓON es la sumatoria de la producción de los días laborados.
NIVEL PRODUCTIVIDAD es un comentario que indica DEFICIENTE, BUENO o EXCELENTE si el TOTAL PRODUCCIÓON es menor, es igual o es mayor al nivel de productividad respectivamente.
TOTAL es el total del TOTAL PRODUCCIÓON de todas las estaciones de trabajo.

(Primero hágalo usted, ...después compare la solución.)

```
Algoritmo ESTACIONES DE TRABAJO
1. Declarar
     Variables
       observacion: Cadena
       hay: Caráacter
       estacion, proDia, totProd, nivProductividad,
       toTotProd: Entero
2. Imprimir encabezado
3. toTotProd = 0
4. Solicitar Nivel de productividad
5. Leer nivProductividad
6. for estacion=1; estacion<=10; estacion++
       a. totProd = 0
       b. Preguntar "¿Hay diadía de produccion (S/N)?"
       c. Leer hay
       d. while hay == 'S'
            1. Solicitar Producción del día
```

```
            2. Leer proDia
            3. totProd = totProd + proDia
            4. Preguntar "¿Hay diadía de Produccion (S/N)?"
            5. Leer hay
      e. endwhile
      f. if totProd < nivProductividad then
            1. observaciónon = "DEFICIENTE"
      g. else
            1. if totProd = nivProductividad then
                  a. observacionobservación = "BUENO"
            2. else
                  a. observacionobservación = "EXCELENTE"
            3. endif
      h. endif
      i. Imprimir estacion, totProd, observacionobservación
      j. toTotProd = toTotProd + totProd
7. endfor
8. Imprimir toTotProd
9. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C544.C y
Programa en Java: Estaciones1.java

Explicación:

1. Se declaran las variables
2. Imprime encabezado
3. Inicia toTotProd en 0
4. Solicita Nivel de productividad
5. Se lee en nivProductividad
6. Inicia ciclo for desde estacionestación = 1 hasta 10 con incrementos de 1
   a. Inicia totProd en 0
   b. Pregunta "¿Hay diadía de produccionproducción (S/N)?"
   c. Lee en hay la respuesta
   d. Inicia ciclo while mientras hay == 'S'; entra al ciclo
      1. Solicita Producción del día
      2. Lee en proDia
      3. Incrementa totProd con proDia
      4. Pregunta "¿Hay diadía de produccionproducción (S/N)?"
      5. Lee en hay la respuesta
   e. Fin del ciclo while
   f. Si totProd < nivProductividad entonces
      1. Coloca "DEFICIENTE" en observacionobservación
   g. Si no
      1. Si totProd = nivProductividad entonces
         a. Coloca "BUENO" en observacionobservación
      2. Si no

a. Coloca “EXCELENTE” en ~~observacion~~observación
3. Fin del if
h. Fin del if
i. Imprime ~~estacion~~estación, totProd, ~~observacion~~observación
j. Incrementa toTotProd con totProd
7. Fin del for
8. Imprime toTotProd
9. Fin del algoritmo

Ejercicio 5.3.3.6

Similar al ejercicio anterior, con la diferencia de que ahora se tienen varias estaciones de trabajo no exacto. Por cada estación se tiene el número de identificación y la producción de cada uno de los días de la semana que trabajó. Elaborar un algoritmo que lea los datos e imprima el reporte ya conocido, pero al final que imprima la estación más productiva (suponiendo que no habrá dos estaciones con el total producción igual) y cuánto fue su producción.

(Primero hágalo usted, ~~...~~después compare la solución.)

```
Algoritmo ESTACIONES DE TRABAJO
1. Declarar
     Variables
       observacion: Cadena
       otra, hay: Car~~á~~acter
       estacion, proDia, totProd, nivProductividad,
       toTotProd, estacionMayor, prodMayor: Entero
2. Imprimir encabezado
3. toTotProd = 0; prodMayor = 0
4. Solicitar Nivel de productividad
5. Leer nivProductividad
6. do
     a. totProd = 0
     b. Solicitar Número de estación de trabajo
     c. Leer ~~estacion~~estación
     d. Preguntar "¿Hay ~~dia~~día de ~~produccion~~producción
(S/N)?"
     e. Leer hay
     f. while hay == 'S'
          1. Solicitar Producción del día
          2. Leer proDia
          3. totProd = totProd + proDia
          4. Preguntar "¿Hay dí~~i~~a de producci~~ó~~ón (S/N)?"
          5. Leer hay
     g. endwhile
     h. if totProd < nivProductividad then
          1. observaci~~ó~~ón = "DEFICIENTE"
```

```
      i. else
            1. if totProd = nivProductividad then
                  a. observacionobservación = "BUENO"
            2. else
                  a. observacionobservación = "EXCELENTE"
            3. endif
      j. endif
      k. Imprimir estacionestación, totProd,
observacionobservación
      l. if totProd > prodMayor then
            1. prodMayor = totProd
            2. estacionMayor = estacionestación
      m. endif
      n. toTotProd = toTotProd + totProd
      o. Preguntar "¿Hay otra estacionestación (S/N)?"
      p. Leer otra
7. while otra == 'S'
8. Imprimir toTotProd, estacionMayor, prodMayor
9. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C545.C y
Programa en Java: Estaciones2.java

Explicación:

1. Se declaran las variables
2. Imprime encabezado
3. Inicia toTotProd en 0; prodMayor en 0
4. Solicita nivel de productividad
5. Lee en nivProductividad
6. Inicia ciclo do
   a. Inicia totProd en 0
   b. Solicita número de estación de trabajo
   c. Lee en estacionestación
   d. Pregunta "¿Hay diadía de produccionproducción (S/N)?"
   e. Lee en hay la respuesta
   f. Inicia ciclo while mientras hay == 'S'; entra al ciclo
      1. Solicita producción del día
      2. Lee en proDia
      3. Incrementa totProd con proDia
      4. Pregunta "¿Hay diadía de produccionproducción (S/N)?"
      5. Lee en hay la respuesta
   g. Fin del ciclo while
   h. Si totProd < nivProductividad entonces
      1. Coloca en oservacionobservación "DEFICIENTE"
   i. Si no
      1. Si totProd = nivProductividad entonces

a. Coloca en ~~oservacion~~observación “BUENO”
2. Si no~~i~~
a. Coloca en ~~oservacion~~observación “EXCELENTE”
3. Fin del if
j. Fin del if
k. Imprime ~~estacion~~estación, totProd, ~~oservacion~~observación
l. Si totProd > prodMayor entonces
1. Coloca totProd en prodMayor
2. Coloca ~~estacion~~estación en estacionMayor
m. Fin del if
n. Incrementa toTotProd con totProd
o. Pregunta “¿Hay otra ~~estacion~~estación (S/N)?”
p. Lee en otra la respuesta
7. Fin del do…while mientras otra == ‘S’ vuelve al do, si no se sale del ciclo
8. Imprime toTotProd, estacionMayor, prodMayor
9. Fin del algoritmo

Ejercicio 5.3.3.7

Elaborar un algoritmo que imprima los valores de X, Y~~,~~ y Z. Y y Z son valores dependientes de X; X deberá tomar valores desde 0.5 hasta 10 con incrementos de 0.5. Para cada valor de X, los valores de Y y Z se calculan con las siguientes ecuaciones:

$Y = 3X^2 + 7X - 15$
$Z = Y - 2X^2$

Debe imprimir el reporte:

VALORES DE X Y Z

X	Y	Z
99.99	99.99	99.99
99.99	99.99	99.99
--		
--		
99.99	99.99	99.99

Nota:
~~É~~Este es un problema tipo for porque se conoce cuántas veces se debe repetir el ciclo, sin embargo, debido a que hay lenguajes en los que la variable de control del for sólo funciona con valores tipo entero~~,~~, no es posible resolverlo con for, la solución debe hacerse con while o con do…while.

Con formato: Diseño: Claro (Gris 12,5%)

Solución con while:

(Primero hágalo usted, ~~...~~después compare la solución.)

```
Algoritmo VALORES X Y Z
1. Declarar
     Variables
       x, y, z: Real
2. Imprimir encabezado
3. x = 0
4. while x < 10
      a. x = x + 0.5
      b. y = 3 * Potencia(x,2) + 7*x - 15
      c. z = y – 2 * Potencia(x,2)
      d. Imprimir x, y, z
5. endwhile
6. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C546.C y
Programa en Java: ValoresXYZ1.java

Explicación:
1. Se declaran las variables
2. Imprime encabezado del reporte
3. Se inicia x en 0
4. Inicia ciclo while pregunta si x es menor que 10, si es así, entra al ciclo
   a. Incrementa x en 0.5
   b. Calcula y
   c. Calcula z
   d. Imprime x, y, z
5. Fin del ciclo while
6. Fin del algoritmo

Solución con do…while:

(Primero hágalo usted, …después compare la solución.)

```
Algoritmo VALORES X Y Z
1. Declarar
     Variables
       x, y, z: Real
2. Imprimir encabezado
3. x = 0
4. do
     a. x = x + 0.5
     b. y = 3 * Potencia(x,2) + 7*x - 15
     c. z = y – 2 * Potencia(x,2)
     d. Imprimir X, Y, Z
5. while x < 10
```

```
6. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C547.C y
Programa en Java: ValoresXYZ2.java

Explicación:
1. Se declaran las variables
2. Imprime encabezado
3. Se inicia x en 0
4. Inicia ciclo do
   a. Se incrementa x en 0.5
   b. Se calcula y
   c. Se calcula z
   d. Imprime x, y, z
5. Fin ciclo while si x es menor que 10 regresa al do; si no se sale del ciclo
6. Fin del del algoritmo

Solución con for:

Para lenguajes que aceptan variables tipo real en la variable de control del ciclo for:

(Primero hágalo usted, ...después compare la solución.)

```
Algoritmo VALORES X Y Z
1. Declarar
     Variables
       x, y, z: Real
2. Imprimir encabezado
3. for x=0; x<=10; x=x+0.5
       a. y = 3 * Potencia(x,2) + 7*x - 15
       b. z = y – 2 * Potencia(x,2)
       c. Imprimir x, y, z
4. endfor
5. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C548.C y
Programa en Java: ValoresXYZ3.java

Explicación:
1. Se declaran las variables
2. Imprime encabezado
3. Inicia ciclo for desde x = 0 hasta 10 con incrementos de 0.05
   a. Calcula y
   b. Calcula z

c. Imprime x, y, z
4. Fin del ciclo for
5. Fin del algoritmo

Ejercicio 5.3.3.8

Elaborar un algoritmo para enlistar los números pares entre 0 y 20.

(Primero hágalo usted, ...después compare la solución.)

```
Algoritmo IMPRIME PARES 0-20
1. Declarar
     Variables
       i: Entero
2. i = 0
3. while i <= 20
      a. Imprimir i
      b. i = i + 2
4. endwhile
5. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C549.C y
Programa en Java: Pares.java

Explicación:
1. Se declaran las variables
2. Inicia i en 0
3. Inicia ciclo while se pregunta si i <= 20, si es así, entra al ciclo
   a. Imprime i
   b. Incrementa i en 2
4. Fin del ciclo while
5. Fin del algoritmo

Ejercicio 5.3.3.9

Elaborar un algoritmo capaz de leer un valor entero positivo y que determine e imprima si este valor es par o impar.

(Primero hágalo usted, ...después compare la solución.)

```
Algoritmo PAR IMPAR
1. Declarar
     Variables
       numero, residuo: Entero
2. Solicitar el número
3. Leer numeronúmero
```

```
4. residuo = nú~~u~~mero
5. while residuo > 1
      a. residuo = residuo - 2
6. endwhile
7. if residuo == 0 then
      a. Imprimir ~~numero~~número, "ES PAR"
8. else
      a. Imprimir ~~numero~~número, "ES IMPAR"
9. endif
10. Fin
```

En la zona de descarga de la Web del libro, están disponibles:
Programa en C: C550.C y
Programa en Java: ParImpar.java

Explicación:

1. Se declaran las variables
2. Se solicita el número
3. Se lee en ~~numero~~número
4. Se inicia residuo con -n~~u~~úmero
5. Inicia ciclo while compara si residuo > 1~~;~~, si se cumple, entra al ciclo
   a. Le resta (quita) 2 a residuo
6. Fin del ciclo while
7. Si residuo == 0 entonces
   a. Imprime nú~~u~~mero, "ES PAR"
8. Si no (else)
   a. Imprime nú~~u~~mero, "ES IMPAR"
9. Fin del if
10. Fin del algoritmo

Nota:
Este problema sólo puede resolverse con while porque la repetición que se presenta puede ir desde 0 (cero) hasta N veces.

Con formato: Diseño: Claro (Gris 12,5%)

Con formato: Fuente: (Predeterminado) Arial, 12 pto

Página 22: [1] Con formato ... **06/03/2010 11:03:00 a.m.**

Español (alfab. internacional)

Página 22: [2] Con formato ... **06/03/2010 11:03:00 a.m.**

Español (alfab. internacional)

Página 22: [3] Con formato ... **06/03/2010 11:03:00 a.m.**

Español (alfab. internacional)

Página 22: [4] Con formato ... **06/03/2010 11:03:00 a.m.**

Español (alfab. internacional)

Página 22: [5] Con formato ... **06/03/2010 11:07:00 a.m.**

Fuente: (Predeterminado) Arial, 12 pto, Disminuido 12 pto

Página 22: [6] Con formato ... **06/03/2010 11:09:00 a.m.**

Español (alfab. internacional)

Página 22: [7] Con formato ... **06/03/2010 11:09:00 a.m.**

Fuente: (Predeterminado) Arial, 12 pto, Disminuido 12 pto

Página 22: [8] Con formato ... **06/03/2010 11:09:00 a.m.**

Español (alfab. internacional)