UNIAD TRES ESTRUCTURAS

1. Estructuras Repetitivas
   1.1 estructuras de repetición o iteración
      1.1.1 un contador
      1.1.2 un acumulador

2. Tipos de estructuras Repetitivas
   2.1 Estructuras de mientras que…. haga
   2.2 Estructura Repita / hasta que

3. Comparaciones de los bucles Whil Repeat
   3.1 Estructura Para.. hasta…haga (o desde..hasta… haga)

4. Estructura Caso o Case

1 ESTRUCTURAS REPETITIVAS.

Las estructuras que repiten una secuencia de instrucciones un número determinado de veces se denominan Bucles y se denomina Iteración al hecho de repetir la ejecución de una secuencia de acciones. Entre las estructuras repetitivas se encuentran:

- Mientras (while)
- Repetir (repeat)
- Desde (for)

1.1. Estructuras de repetición o interacción

Las instrucciones de repetición o interacción sirven para sacar varios valores o resultados. Para ello es necesario un contador y un acumulador.

1.1.1 Un contador

Es una variable que sirve para llevar una cuenta con incrementos o decrementos constates. Ejemplo. m = m + 1.

Analizando el ejemplo, se observa que el incremento es 1 cada vez que se pasa por dicha línea. También se ve que al lado y lado se escribe coloca o escribe la misma variable, esto se hace con el fin de llevar la cuenta el valor anterior. La variable m, debe poseer un valor inicial para poder empezar a ejecutarse.

Si m, inicialmente vale 3 es decir m=3 y el contador es m = m + . entonces m, tomara el valor de 4 (3+1); cuando vuelva a pasar por l línea m, será igual a 5, la tercera vez tomara el valor de 6 y así sucesivamente.

Ejemplo: C = C + 10

Inicialmente C es igual a cero (C = 0)

El incremento es de 10 cada vez que se pase por el contador. Entonces la primera vez vale 10, la segunda 20, la tercera 30 y así sucesivamente.

1.1.2 Un acumulador

Es una variable que sirve para guardar y acumular valores que pueden ser diferentes cada vez. Es una variable en la que se puede ir calculando la suma de los valores que tome otra variable dentro del algoritmo.

Ejemplo: Acum = Acum + V

Tot = Tot + Num

El nombre del acumulador se escribe al lado y lado el signo = porque es necesario que cada vez que se pase por dicha línea, se comienza por el valor que se había quedado antes. El valor que se suma o incrementa puede ser diferente cada vez. Para poder efectuar la operación de acumulación es necesario que tanto el acumulador como el valor que se está acumulando, hayan tenido antes un valor inicial.

2 Tipos de estructuras repetitivas o interactivas

La estructura algorítmica mientras comúnmente conocida como while, es la estructura adecuada para utilizar en un ciclo cuando no sabemos el número de veces que éste se ha de repetir. Dicho número depende de las proposiciones dentro del ciclo. Esta estructura permite que se repita una acción o un conjunto de acciones, en tanto cierta condición se mantenga verdadera. La estructura while evalúa primero la condición, si se cumple se ejecuta una acción o conjunto de acciones; si no se cumple, no entra al ciclo. Por lo tanto esta estructura se repite cero o más veces.

2.1 Estructura mientras que….. Haga…

Formato:

Hacer PI = proposición inicial
Mientras PI es verdadera repetir
•
•
proceso
•
•
•
Hacer PI = modificación de PI
{ Fin del ciclo }

Donde:

PI.- proposición inicial, si el valor de PI es falso, entonces el ciclo no se ejecuta.

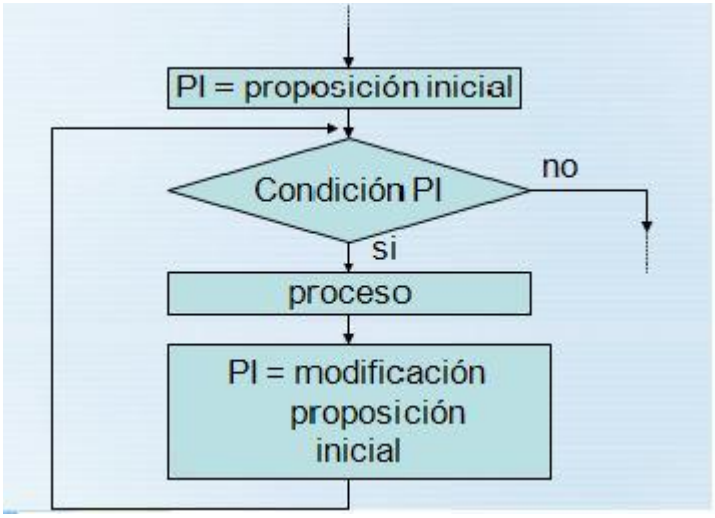


Imagen1.mientras que.

Ejemplo1. Leer e imprimir o escribir 5 nombres de personas.

a. Análisis de la solución

El ejercicio pide que se lea un nombre y se imprima, luego el segundo nombre y se imprima y así sucesivamente hasta obtener los 5 nombres de personas. Por lo tanto se necesita una variable que sirva para leer y otra que lleve la cuenta de los nombres leídos y mostrados.

b. Definición e variables

E, S ⟶ NOM = sirve para leer e imprimir el nombre de 5 personas.

P ⟶ CN = sirve para llevar la cuenta de las personas leídas. (Contador)

La variable NOM es e entrada porque sirve para leer el nombre de las personas y es de salida porque sirve para mostrar el nombre que guarda.

La variable CN es una variable de proceso, porque va contando los nombres leídos.

c. Diagrama de flujo

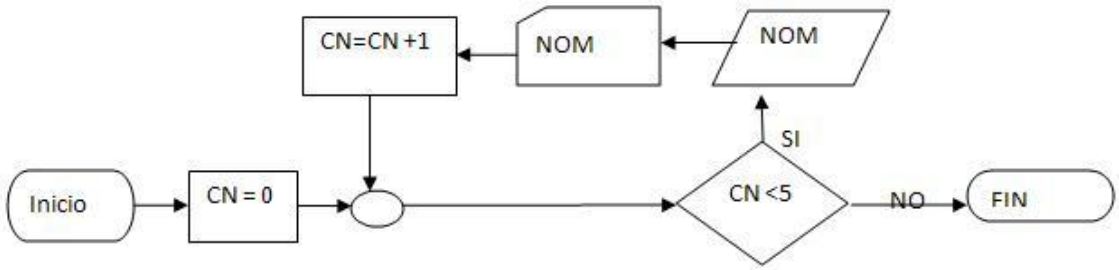


Imagen2. Diagrama de flujo mientras que

Analizando el diagrama se ve:

- El contador de personas se inicializa o se le asigna un valor de 0, porque no se ha leído ningún nombre.
- Los pasos de la estructura que es necesario que se repitan están dentro de la estructura
- El contador se aumenta en uno (1) cada vez que se lee o escribe un nombre.
- Cuando CN sea = 5 buscara el fin del programa y el proceso.

d. Algoritmo

```
Inicio
   CN = 0
    Mientras  que N < 5 haga
       Lea NOM
       Escriba NOM
      CN + N + 1
  Fin_mq
Fin- algoritmo
```

Ejemplo 2. Calcular el valor de N aéreas de triángulos, además imprimir la sumatoria y el promedio de las aéreas calculadas.

a. Análisis de la solución

Se saca la fórmula del área que es Área = base * altura / 2. Se definen las variables para calcular el área, un contador y la variable N que indica hasta donde ira o cuantas veces se repite el ciclo o bucle. También es necesario acumular el valor de cada área calculada y posteriormente sacar el promedio, por lo cual se necesitan otras 2 variables.

b. Definición de variables

N. sirve para determinar el número de áreas que se desean calcular
B. sirve para leer la base del triangulo
H. sirve para leer la altura de triangulo
A . sirve para calcular el valor del área del triangulo
CA. sirve para contar las áreas calculadas
ACA sirve para calcular e imprimir la sumatoria de las áreas calculadas
PROM sirve para calcular e imprimir el promedio de las áreas calculadas

c. Diagrama de Flujo.

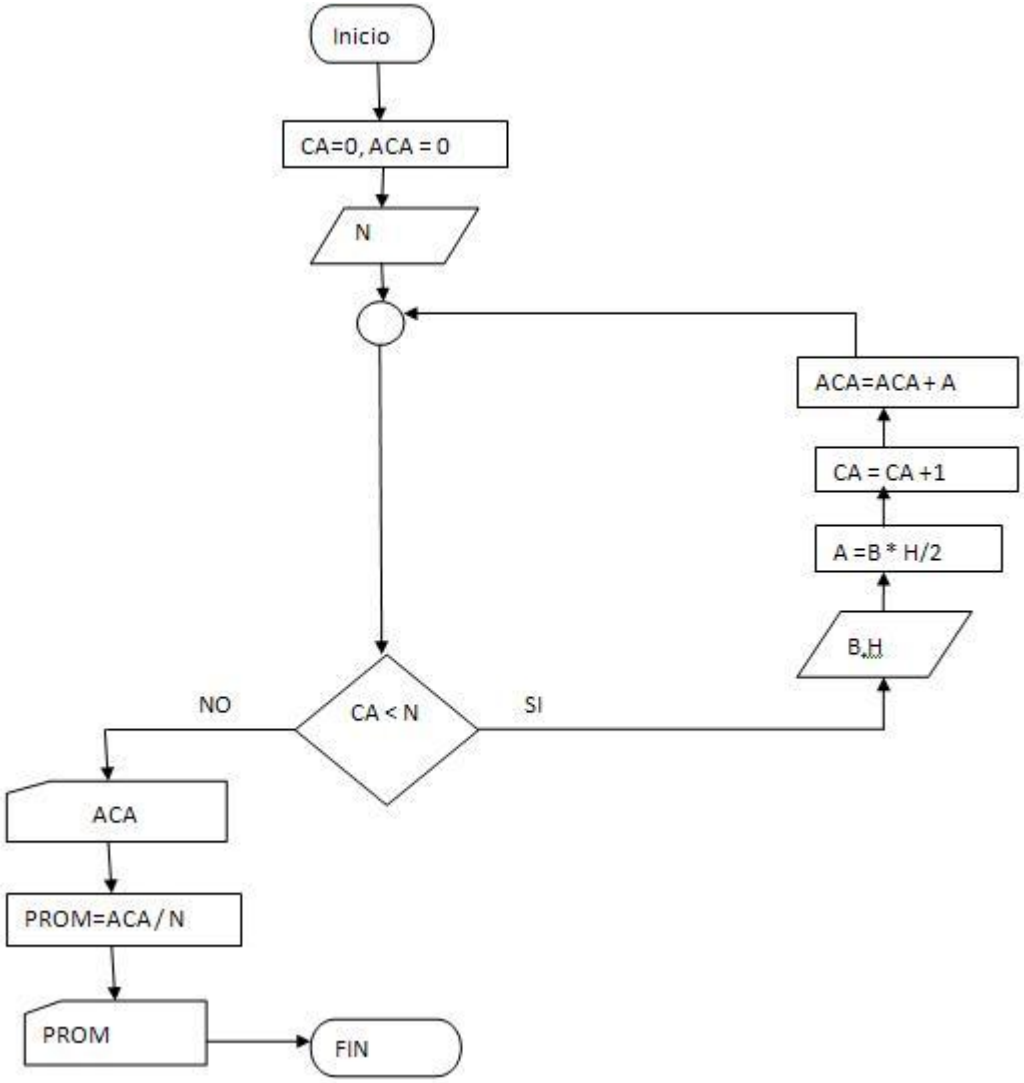


Imagen3.digrama ejemplo2

d. Prueba de escritorio

CA	ACA	N	B	H	A	PROM
0	0	5	2	4	4	42.7
1	4		7	12	42	
2	46		3	5	7.5	
3	53.5		12	20	120	

Imagen4.prueba de escritorio

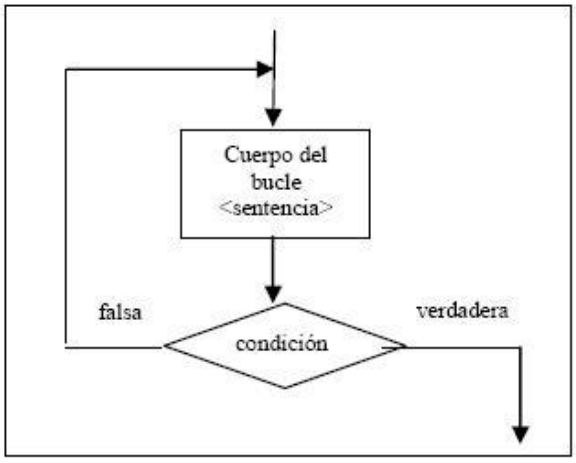
e. Algoritmo

```
Inicio
    CA = 0,  ACA = 0
    Lea N
    Mientras  que Ca < N haga
      Lea  B,H
      A=  B * H / 2
      CA =  CA + 1
      ACA   =  ACA  +  A
     Fin mq
     Escriba  ACA
     PROM =  ACA / CA
     Escriba  PROM
Fin algoritmo
```

2.2 Estructura Repita / hasta que……

Es otras de las estructuras de repetición y se utiliza para calcular más de un valor. Una variante de la sentencia while es la sentencia repeat. Una de las características de los bucles-do es que la condición se evalúa al principio de cada iteración. En particular, si la condición es falsa cuando la sentencia comienza, entonces el bucle no se ejecuta nunca. La sentencia repeat especifica un bucle condicional que se repite hasta que la condición se hace verdadero. Tal bucle se denomina bucle repeat-until.

El diagrama de flujo y el pseudocódigo




```
Repetir
                Sentencia
Hasta_que condición
```

Condición expresión lógica o relacional

Imagen5.repita

Se repite hasta que o hasta cuando se cumpla la condición, se ejecutan todas las condiciones que se encuentran dentro de la estructura es decir, todo lo que encuentre por Falso o No; cundo la condición no se cumpla ejecutará las instrucciones que se encuentran por el camino opuesto.

Ejercicio. Leer 10 nombres de partes del computador e imprimirlos o visualizarlos.

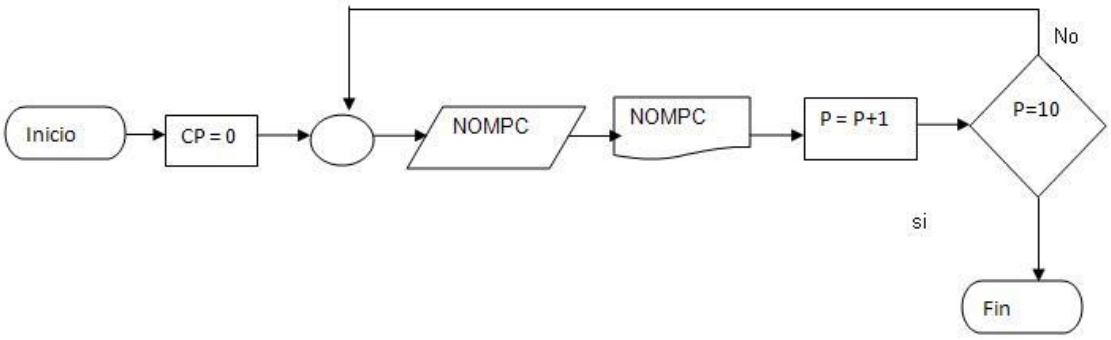
a. Análisis de la solución.

Con solo definir una variable para leer un nombre de la parte del computador, ese mismo sirve para leer todos los datos pedidos.

b. Definición de Variables

NOMPC Sirve para leer e imprimir el nombre de las 10 partes del computador
CP. Sirve para leer los nombres de las partes del computador leídos y que se van imprimiendo.

c. Diagrama de Flujo



Imgen6.ejemplo repita.

d. Algoritmo

```
Inicio
        CP  = 0
         Repita
            Lea  NOMPC
            Escriba  NOMPC
            CP  =  CP + 1
        Hasta que CP = 10
Fin algoritmo.
```

3.. Comparaciones de los bucles Whil Repeat

Los bucles while y repeat son complementados en su actuación. La elección de una sentencia u otra dependerá del diseño del problema por parte del programador. Si bien con la sentencia while se pueden escribir casi todos los algoritmos repetitivos, en muchas ocasiones la sentencia repeat facilita la escritura de algoritmos. En general, el cuerpo del bucle repeat se ejecuta siempre.

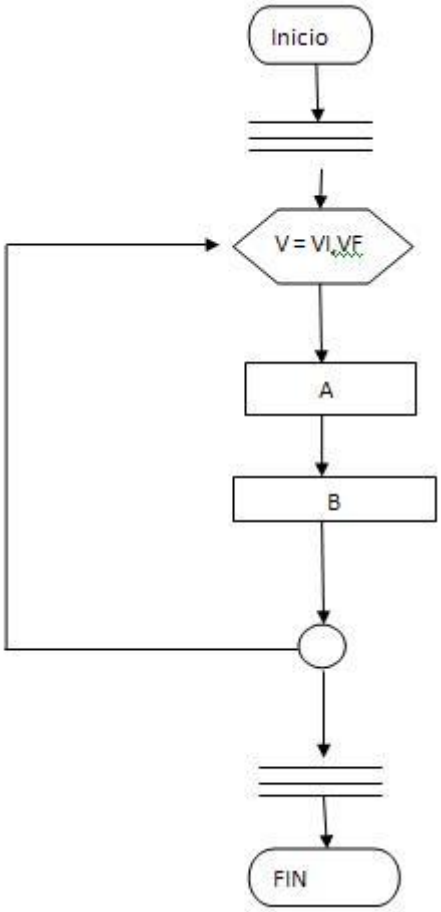
Comparación de while y repeat.

Bucles WHILE	Bucles REPEAT
1. La condición (expresión lógica) se verifica antes de que se ejecute el cuerpo del bucle.	1. La condición bucle se verifica después que el cuerpo del bucle se ha ejecutado.
2. Como resultado de (1), el cuerpo del bucle puede no ser ejecutado (si la condición es falsa.	2. Como resultado de (1), el cuerpo del bucle se ejecutara al menos una vez (no importa cual será el valor de la condición)
3. Como resultado de (1), las variables de la condición deben haber sido inicializadas antes de alcanzar la sentencia while, de modo que la condición puede ser verificada.	3. Como resultado de (1), las variables de la condición no necesitan ser inicializa-das antes de alcanzar al sentencia repeat A esas variables se le pueden asignar valores en el cuerpo del bucle que se ejecutaran antes de que se verifique la condición.
4. Si la condición es verdadera, el cuerpo del bucle se ejecutara y continuará el bucle.	4. Si la condición es verdadera, el cuerpo del bucle se habrá ejecutado pero se detiene el bucle.
5. Para evitar un bucle infinito, debe asegurarse que la condición contenga una variable cuyo valor se modifique en el cuerpo del bucle, pasando a tomar el valor falso.	5. Para evitar un bucle infinito, debe ase-gurarse que la condición contenga una variable cuyo valor se modifique en el cuerpo del bucle, pasando a tomar el valor verdadero.

Imagen7. While repeat.

3.1 Estructura Para… hasta… haga ….. (o desde…hasta….haga)

La estructura repetitiva Para… es aquella en que numero de iteraciones del bucle o ciclo es determinado directamente, por lo tanto se sabe en qué valor debe comenzar y en qué valor terminar.



Imgen8.para hasta

La estructura para o desde…. Ejecuta las instrucciones el ciclo un número determinado de veces y controla de manera automática el número de repeticiones o pasos que se tienen que llevar a cabo. Cuando termina de ejecutar las instrucciones automáticamente sale de la estructura y va a buscar las que se encuentran por fuera y a continuación de ella, hasta encontrar el fin del algoritmo, diagrama o programa.

Ejercicio. Leer 7 nombres de empleados e una fabrica con su respectiva cantidad de producción.

a. Análisis de la solución.

Se necesita una variable para leer el nombre del empleado, además se debe conocer la cantidad de producción generada por empleado y también es necesario controlar el fin del proceso, por lo tanto se debe ir contando los empleados leídos y cuando se llegue al fin se detiene.

b. Definición de variables

NOMB sirve para leer el nombre de cada uno de los empleados y visualizar el contenido de las variables.
PROD sirve para leer la producción de cada uno de los empleados y mostrar cuanta fu su producción.
C. sirve para contar los datos leídos es decir el nombre de las personas y su producción.

c. Diagrama de Flujo

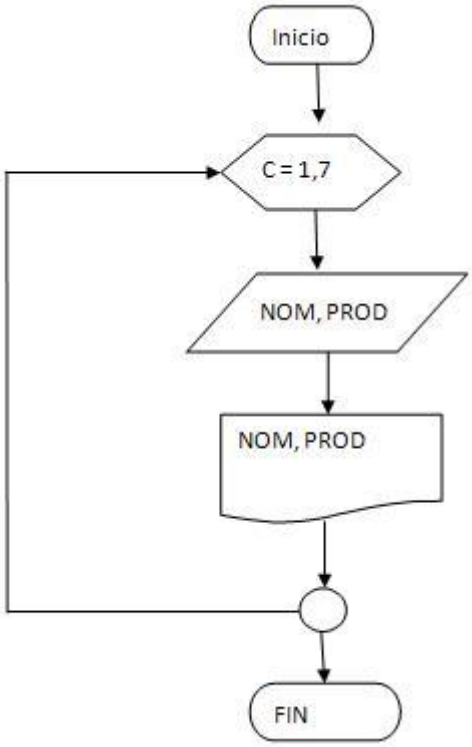


imagen9.diagrama ejemplo para hasta

d. Algoritmo

```
Inicio
   Para  C = 1 hasta 7 haga
         Lea  NOM, PROD
         Escriba  NOM, PROD
   Fin para
Fin algoritmo.
```

5. Estructura Caso o Case

La estructura Caso o Case es una estructura que sirve para seleccionar, entre muchas alternativas, una a la vez, por eso se considera como una estructura múltiple. Se evalúa el valor de la variable de selección o selector y dependiendo del valor asignado se escoge entra varios caminos solo uno.

- Para poder utilizar esta estructura se necesita que existan por lo menos dos alternativas o caminos.
- La etiqueta puede ser de tipo: entero, carácter char, subserie, booleano o enumerado.
- La variable que se utilice como selector deberá ser del mismo tipo de los datos asignados en cada etiqueta.

Ejemplos: 1. Leer un carácter y determinar por medio de un mensaje si es un número entre 0 y 9 o una letra minúscula o mayúscula.

a. Análisis de la solución

Se necesitan 3 etiquetas: una donde se encuentren los números e 0 a 9, otras donde aparezcan las letras minúsculas de la A… Z , y la otra donde se registren las letras mayúsculas de la A… Z. también se debe definir la variable con la cual se lee el carácter.

b. Definición de variables.

CARACT. Sirve para leer el carácter y determinar si es un digito entre 0 y 9 o si es una letra mayúscula o minúscula.

c. Diagrama de Flujo

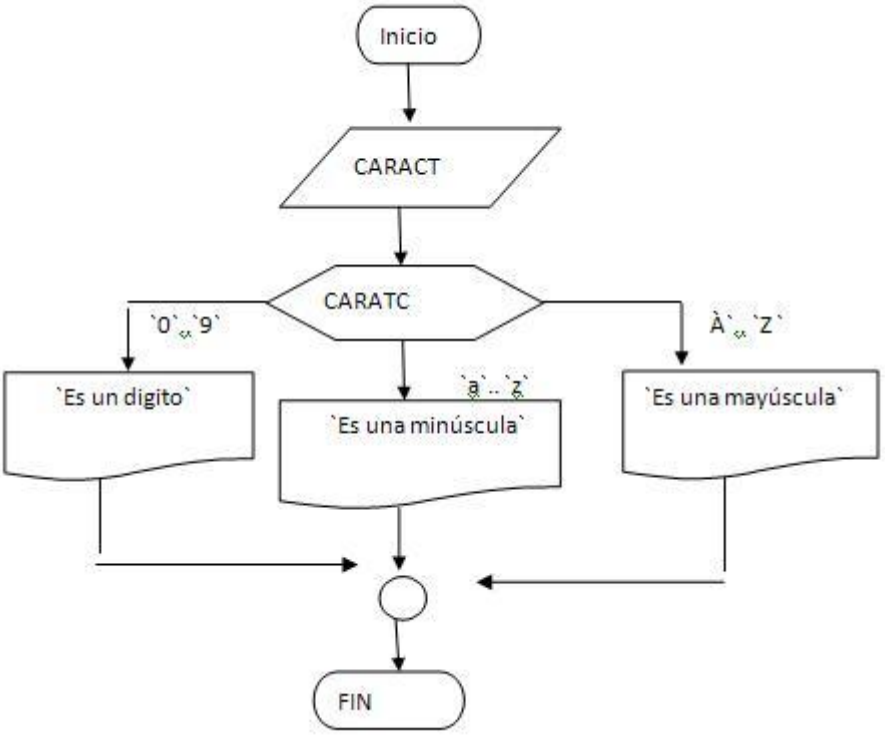


Imagen10. Diagrama caso.

d. Algoritmo.

```
Inicio
   Lea  CARACT
   Según se CARACT  haga
    `0 `.. `9 ` imprima `Es un digito `
    `a `.. `z ` imprima `Es una minúscula `
    `A `.. `Z `imprima `Es una mayúscula
  Fin según
Fin algoritmo.
```

Ejemplo 2. Mostrar en romano el equivalente de los números arábigos 5,10,50,100.

a. Análisis de la solución.

En cada etiqueta ira un número en arábigo y por allí se imprimirá un mensaje diciendo cual es el correspondiente en romano; el número en arábigo servirá e etiqueta.

b. Definición de variables

NUMERO. Sirve para leer el número en arábigo y determinar qué camino o etiqueta seguir.

c. Diagrama de flujo.

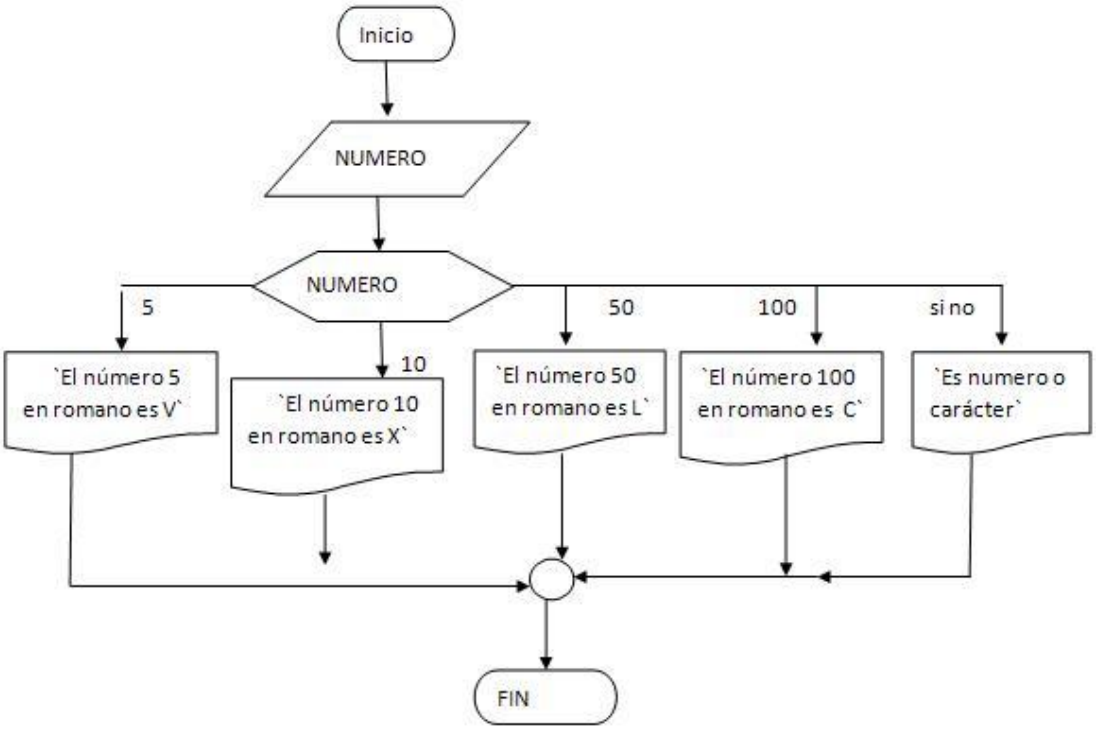


imagen11.diagrama ejercicio caso

d. Algoritmo

```
Inicio
    Lea  NÚMERO haga
    Según sea NÚMERO haga
     5      Imprima  `El número 5 en romano es V`
     10    Imprima  `El número 10 en romano es X`
     50    Imprima  `El número 50 en romano es L`
     100  Imprima  `El número 100 en romano es C`
      Si no
           Imprima  `Es otro número o carácter `
   Fin según sea
Fin algoritmo
```