

Calcular la longitud de una circunferencia dado el radio de la misma.

NOTAS:

- Se hace el cálculo para una sola circunferencia.
- Observe que la pregunta $rad > 0$ debe estar antes del cálculo. En caso de no cumplirse provoca un salto hacia atrás.
- No se emplea una variable para almacenar el perímetro pues no se usará posteriormente.

Análisis del problema:

Longitud Circunf

Necesidades (declaraciones)

1 constante (π)

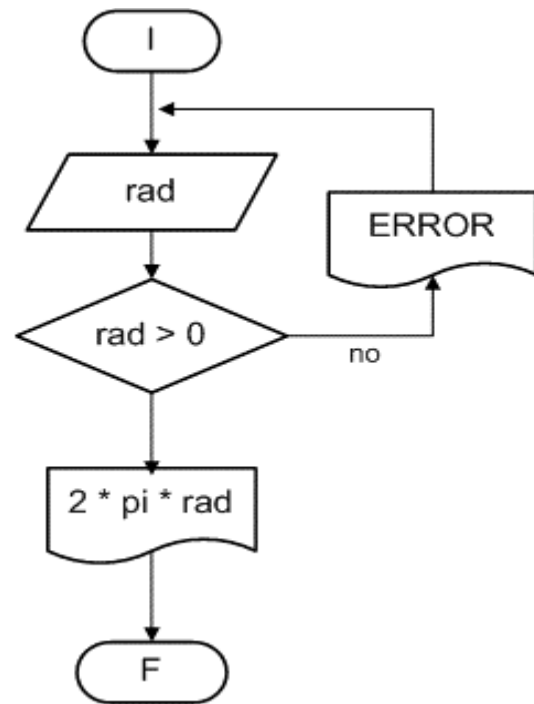
1 variable real (para entrar el radio)

Proceso: $2 * \pi * rad$

Precondiciones: radio positivo

Postcondiciones: no hay

Salidas: perímetro ($2 * \pi * rad$)



Recuerde que el Análisis más el Diagrama de Bloques conforman el Algoritmo.

```
Program LongCircunf;
label 1;
const
  pi=3.14159;
var
  radio:real;
begin
  1: writeln("Teclee el valor del radio ");
    readln(radio);
    if radio<=0 then
      begin
        writeln(' El radio debe ser positivo'); goto 1;
      end;
    writeln(' Longitud: ',2*pi*radio:7:3);
    readln;
end.
```

Algunas consideraciones:

- Esta programación está en Pascal. (no visual)
- Observe la necesidad de una etiqueta (label) para dar el salto hacia atrás. La misma se declara al igual que constantes y variables.
- Se pregunta $[radio \leq 0]$ y no $[radio > 0]$ como en el diagrama de bloques, porque es exactamente lo contrario (**p: radio > 0; ~p: radio ≤ 0**) y resulta más orgánico al no tener que utilizar **ELSE**. Vea que si radio es positivo NO entra al **IF** y el programa continúa en la orden inmediata posterior.
- Note que la orden **readln** al final es para poder ver el resultado en pantalla, ya que ella detiene la ejecución del programa en espera de que se pulse la tecla ENTER.

Calcular la longitud de N circunferencias dado el radio de cada una.

NOTAS:

- Se hace el cálculo para una cantidad determinada de circunferencias. Quiere decir que se puede conocer de antemano o en el momento de ejecución la cantidad de repeticiones.
- Observe que la pregunta $\text{tot} > 0$ debe estar antes de comenzar las iteraciones. En caso de no cumplirse provoca un salto hacia atrás.
- Observe que la pregunta $\text{rad} > 0$ debe estar antes de cada cálculo. En caso de no cumplirse provoca un salto hacia atrás.
- No se emplea una variable para almacenar el perímetro pues no se usará posteriormente.

Análisis del problema:

LongNCircunf

Necesidades (declaraciones)

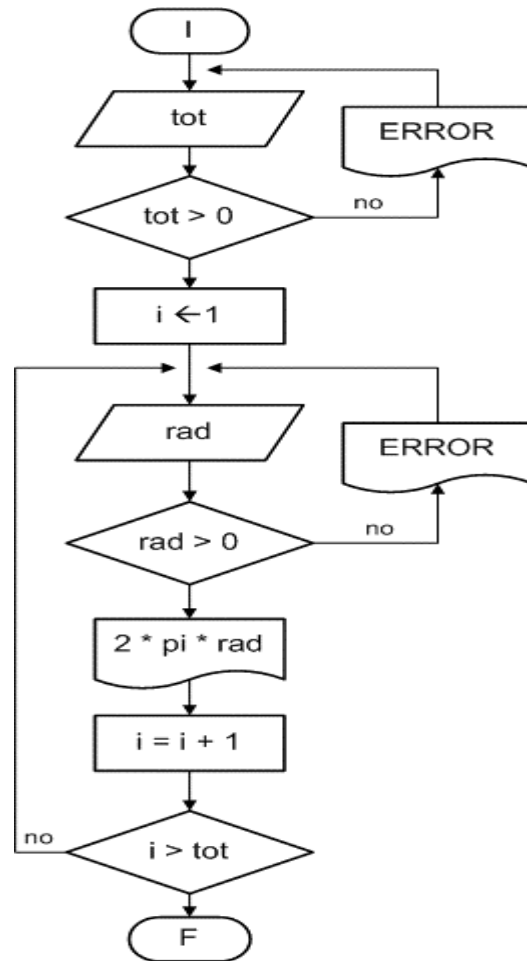
- 1 constante (π)
- 1 variable real (para entrar el radio)
- 2 variables enteras (1 para contar las repeticiones y la otra para el total de ellas)

Proceso: Mostrar cada vez: $2 \cdot \pi \cdot \text{rad}$ hasta completar el total (N).

Precondiciones: total y radio positivos

Postcondiciones: no hay

Salidas: perímetro ($2 \cdot \pi \cdot \text{rad}$) de cada circunf.



Recuerde que el Análisis más el Diagrama de Bloques conforman el Algoritmo.

```
program PerNCircunf;
uses crt;
label 1,2;
const pi=3.14159;
var tot,i:integer;
    rad:real;
begin
  clrscr;
  1: write('Escriba el total de circunferencias: ');
  readln(tot);
  if tot <= 0 then
    begin
      writeln('¡Imposible, debe ser positivo!'); goto 1;
    end;
    { sigue al lado... }
```

```
{ continuación de PerNCircunf... }
for i:=1 to tot do
begin
  2: write('Teclee el valor del radio '); readln(rad);
  if rad <= 0 then
    begin
      writeln('¡Imposible, debe ser positivo!'); goto 2;
    end;
  writeln(' Longitud: ',2*pi*rad:7:3);
end;
readkey;
end.
```

Calcular la longitud de una cantidad indeterminada de circunferencias dado el radio de cada una. Expresar cuántas fueron procesadas.

NOTAS:

- Se hace el cálculo para una cantidad no determinada de circunferencias. Quiere decir que NO se puede conocer de antemano NI en el momento de ejecución la cantidad de repeticiones.
- Se puede aprovechar la propia entrada del radio para establecer -antes que todo- una "marca o bandera" que detendrá el proceso cuando se desee. Dicha "marca" ha de ser un valor (con varias cifras) NO permisible del problema.
- Observe que la pregunta $rad > 0$ debe estar antes de cada cálculo. En caso de no cumplirse provoca un salto hacia atrás. (por otra vía)
- No se emplea una variable para almacenar el perímetro pues no se usará posteriormente.
- Se emplea un contador para responder la segunda parte del problema.

Análisis del problema:

LongNCircunf

Necesidades (declaraciones)

1 constante (π)

1 variable real (para entrar el radio o la marca)

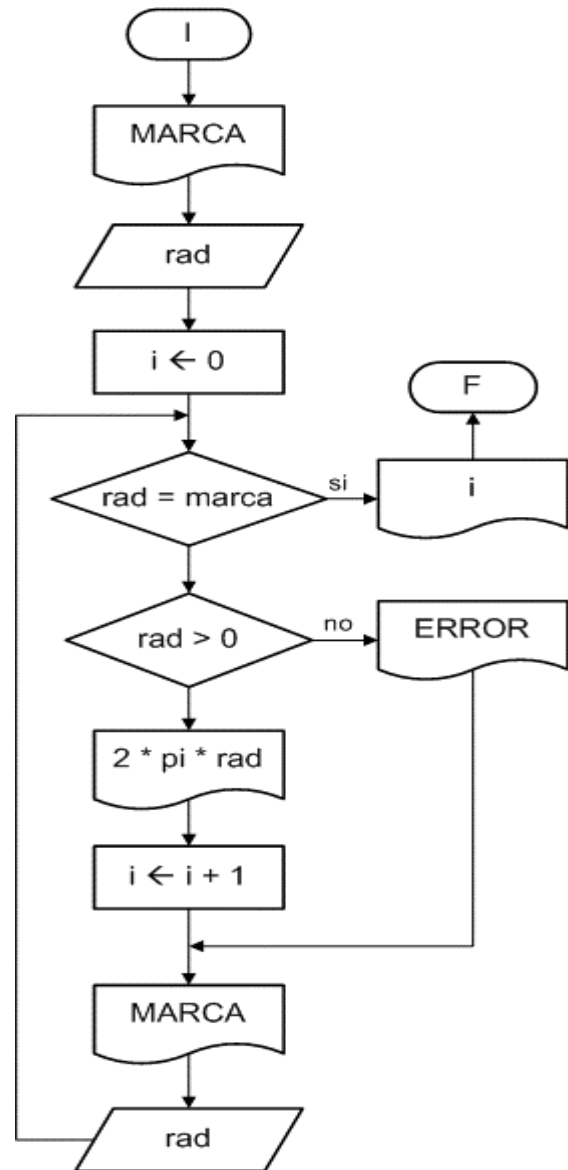
1 variable entera (para contar las repeticiones)

Proceso: Mostrar cada vez: $2 \cdot \pi \cdot rad$ hasta que se decida terminar. También contar cada vez para mostrar al final.

Precondiciones: radio positivo

Postcondiciones: no hay

Salidas: perímetro ($2 \cdot \pi \cdot rad$) de cada circunf.
total de veces (al final)



Recuerde que el Análisis más el Diagrama de Bloques conforman el Algoritmo.

En la página siguiente está un programa (pascal) y una "corrida en seco" (rastreo).

```

Program PerVarCircunf;
uses crt;
const pi=3.14159;
var i:integer;
    rad:real;
begin
  clrscr;
  writeln('Si escribe -6.49 se termina');
  write('Teclee el valor del radio '); readln(rad);
  i := 0;
  while rad <> -6.49 do
  begin
    while rad <= 0 do
    begin
      clrscr;
      writeln('Imposible, debe ser positivo');
      write('Teclee el valor del radio ');
      readln(rad);
    end;
    writeln(' Longitud: ',2*pi*rad:7:3);
    i := i + 1;
    writeln;
    writeln('Si escribe -6.49 se termina');
    write('Teclee el valor del radio ');
    readln(rad);
  end;
  writeln;
  writeln(' Se calcularon ',i,' perímetros. ');
  readkey;
end.

```

Rastreo (corrida en seco)

Marca	rad	i	rad > 0	2*pi*rad	i
-6.49	3	0	si	valor	1
-6.49	-2	1	no	----	1
-6.49	4.2	1	si	valor	2
-6.49	0	2	no	----	2
-6.49	2.5	2	si	valor	3
-6.49	-6.49	3	< fin >		

Algunas consideraciones:

- La marca que se emplea (-6.49) está fuera del rango de valores permisibles y se necesita teclear 5 caracteres; NO hay abandonos "accidentales".
- La estructura WHILE se interpreta así: mientras la condición sea verdadera se repetirá todo su contenido (begin-end;); solamente cuando sea falsa dichas instrucciones no se ejecutarán y pasará a las intrucciones subsiguientes del programa.
- Por la causa anterior suele adoptarse la estrategia de dos lecturas: una para comenzar (solamente se ejecuta una vez) y otra que se repetirá todas las veces que sea necesario; con ello se evitan saltos (goto etiquetas).
- Como se aprecia también son muy útiles para "validaciones de datos" ya que se evitan saltos y las etiquetas.

Calcular la longitud de una cantidad indeterminada de circunferencias dado el radio de cada una. Expresar cuántas tienen una longitud que sobrepase al promedio de las que fueron procesadas.

NOTAS:

- Se hace el cálculo para una cantidad no determinada de ... (igual al anterior)
- Se puede aprovechar la propia entrada del radio para establecer -antes que todo- una "marca o bandera" que detendrá el proceso cuando se desee. Dicha "marca" ha de ser un valor (con varias cifras) NO permisible del problema.
- Observe que la pregunta $\text{rad} > 0$ debe estar antes de cada cálculo. En caso de no cumplirse provoca un salto hacia atrás. (sin etiqueta)
- Si se emplea un tipo de variable para almacenar el perímetro, pues sí se usará posteriormente al tener que comparar cada longitud hallada con el promedio encontrado. (ARREGLO)
- Se emplea un contador y un sumador para responder la segunda parte del problema.

Análisis del problema:

LongNCircunf

Necesidades (declaraciones)

1 constante (π)

2 variables reales (1 para entrar el radio o la marca y 1 para acumular las longitudes)

1 variable de ARREGLO tipo real (para guardar el perímetro de cada circunferencia)

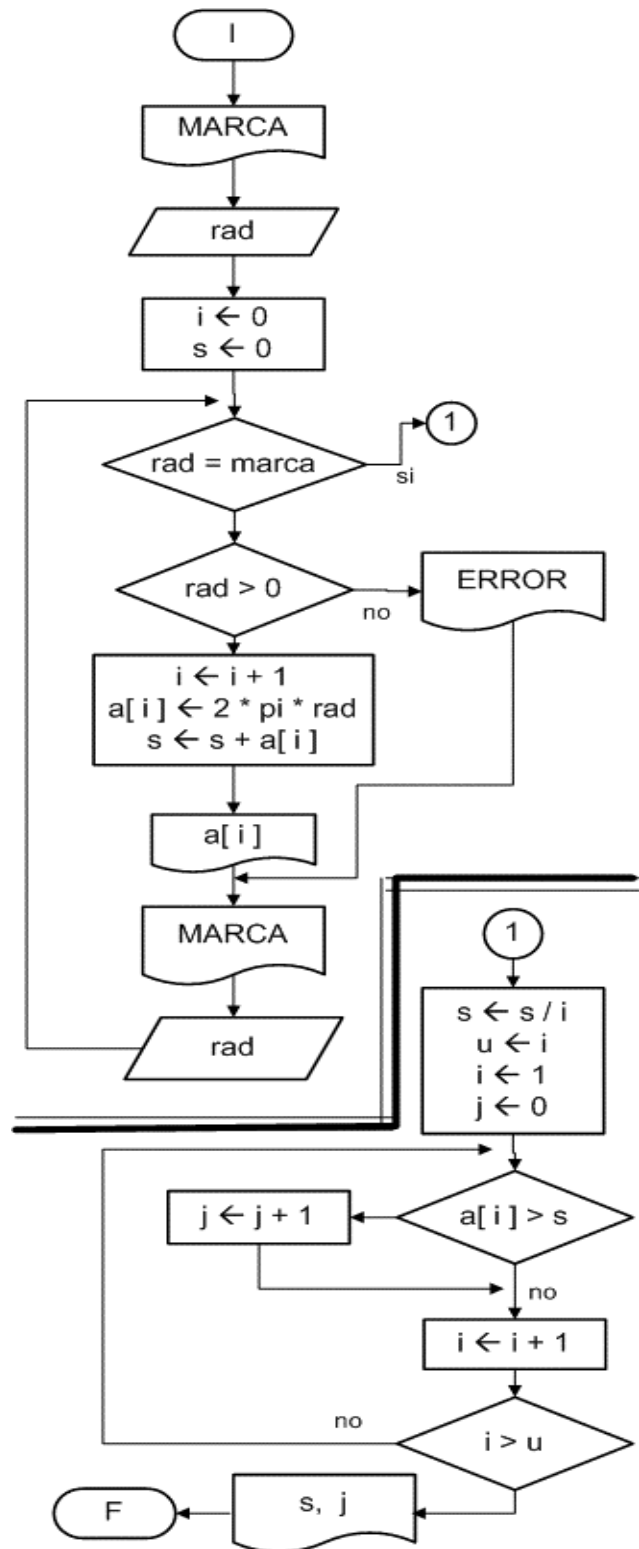
3 variables enteras (2 para contar las repeticiones y los mayores. Otra para el total)

Proceso: Acumular cada vez: $2 \cdot \pi \cdot \text{rad}$ hasta que se decida terminar. También contar cada vez para, al final, hallar el promedio (acum / tot). Después comparar cada longitud almacenada (en el ARREGLO) con el promedio y contar las que sean mayores.

Precondiciones: radio positivo

Postcondiciones: no hay

Salidas: perímetro ($2 \cdot \pi \cdot \text{rad}$) de cada circunf.
promedio de longitudes (al final)
total de las que sobrepasan al promedio hallado (al final)



Recuerde que el Análisis más el Diagrama de Bloques conforman el Algoritmo.

Aquí está un programa (pascal) y una "corrida en seco" (rastreo).

```

Program PerVarCircProm;
uses crt;
const max=50; pi=3.14159;
type lista = array[1..max] of real;
var i,j,u : integer; (*contad. y cant. de elementos*)
    a : lista;
    rad,s : real;
begin
  clrscr;
  i:=0; s:=0;
  writeln('--marca: -5.19 o calcular más de 50');
  write(' Teclee el valor ',i+1,', '); readln(rad);
  while (rad<>-5.19) and (not(i > 50)) do
  begin
    while rad <= 0 do
    begin
      clrscr;
      writeln('Imposible, debe ser positivo');
      write('Teclee el valor del radio ');
      readln(rad);
    end;
    i:= i+1; a[ i ]:= 2*pi*rad; s:= s+a[ i ];
    writeln('Longitud: ',a[ i ]:7:3);
    writeln('...marca.. -5.19 o calcular más de 50');
    write(' Teclee el valor ',i+1,', '); readln(rad);
  end;
  s := s/i; (u:=i); j := 0;
  for i:=1 to u do
  if a[ i ]>s then j:= j+1;
  writeln;
  writeln('prom... ',s:7:3,' encima: ',j);
  readkey;
end.

```

Corrida en seco (rastreo)

<<Primera parte>>

Marca	rad	i	a[i]	s	u	j
-5.19	3.12	1	19.6	19.6	--	--
-5.19	-2					
-5.19	2	2	12.57	32.17	--	--
-5.19	4.06	3	25.51	57.68	--	--
-5.19	2.79	4	17.53	75.21	--	--
-5.19	-5.19	--	--	18.8	4	0

<<Segunda parte>>

--.--	--.--	1	19.6	18.8	4	1
--.--	--.--	2	12.57	18.8	4	1
--.--	--.--	3	25.51	18.8	4	2
--.--	--.--	4	17.53	18.8	4	2