UNIVERSIDAD DE COLIMA

Facultad de ingeniería Mecánica y Eléctrica

Ingeniería en Computación Inteligente

Actividad “Corridas de Prestamos en Go”

Moreno Olmos Luis Miguel

6°B

31/03/2022

La Caja Popular de la Costa Sur, necesita generar corridas para préstamos de dinero en efectivo, bienes muebles y bienes inmuebles. Los intereses y plazos de los préstamos están de acuerdo con la siguiente tabla:

Producto	Plazo mínimo	Plazo máximo	Interés
Efectivo	3 meses	1 año	2% mensual
Bienes muebles	6 meses	2 años	2.5% mensual
Bienes inmuebles	12 meses	5 años	5% mensual por un año disminuyendo en 1% cada año, es decir que si el crédito es a 2 años, el interés será del 4% y así sucesivamente.

- Todos los intereses son sobre saldos insolutos.
- El tipo de pago puede ser fijo o decreciente. En el fijo regularmente se calcula el total de intereses del préstamo y se divide entre el plazo, sumándoselo al capital mensual, quedando un pago fijo mensual. En el decreciente el capital es constante durante el plazo y el interés se va cobrando en su totalidad cada mes.

Se desea:
1) Generar una corrida del crédito indicando todos los datos necesarios para que el cliente vea cómo van disminuyendo sus pagos mensuales (en formato de tabla)
2) Se desea conocer cuánto dinero va a pagar de intereses por todo el crédito.

La aplicación tiene que ser desde la interfaz de comandos o CLI (command-line interface) enviando como argumentos todos los necesarios para realizar el proceso. NO se deben hacer preguntas desde el teclado. Defina usted el formato y orden para obtener los reportes. Use paquetes, funciones, métodos de tipo, slices o lo que necesite para realizar el programa.

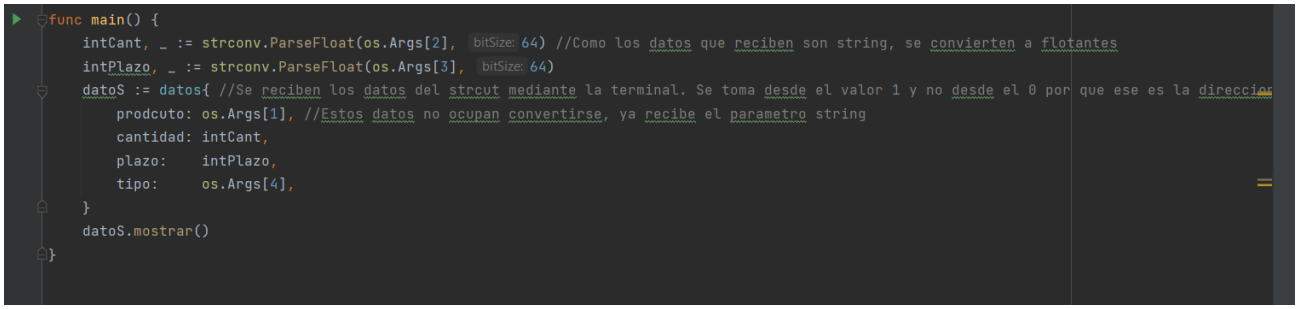
Introducción

Este programa que se presentará este hecho con la finalidad de que imprima corridas de préstamos, estos prestamos pueden ser de tres diferentes productos; El primero es de Efectivo, el segundo es Bienes Muebles y el de tercero de Muebles Inmuebles. Además, cada de que se puede elegir el tipo, también se puede hacer con tipos de cuotas, estas pueden ser fijas o decrecientes.

Desarrollo

EL programa cuenta con:

Un main.



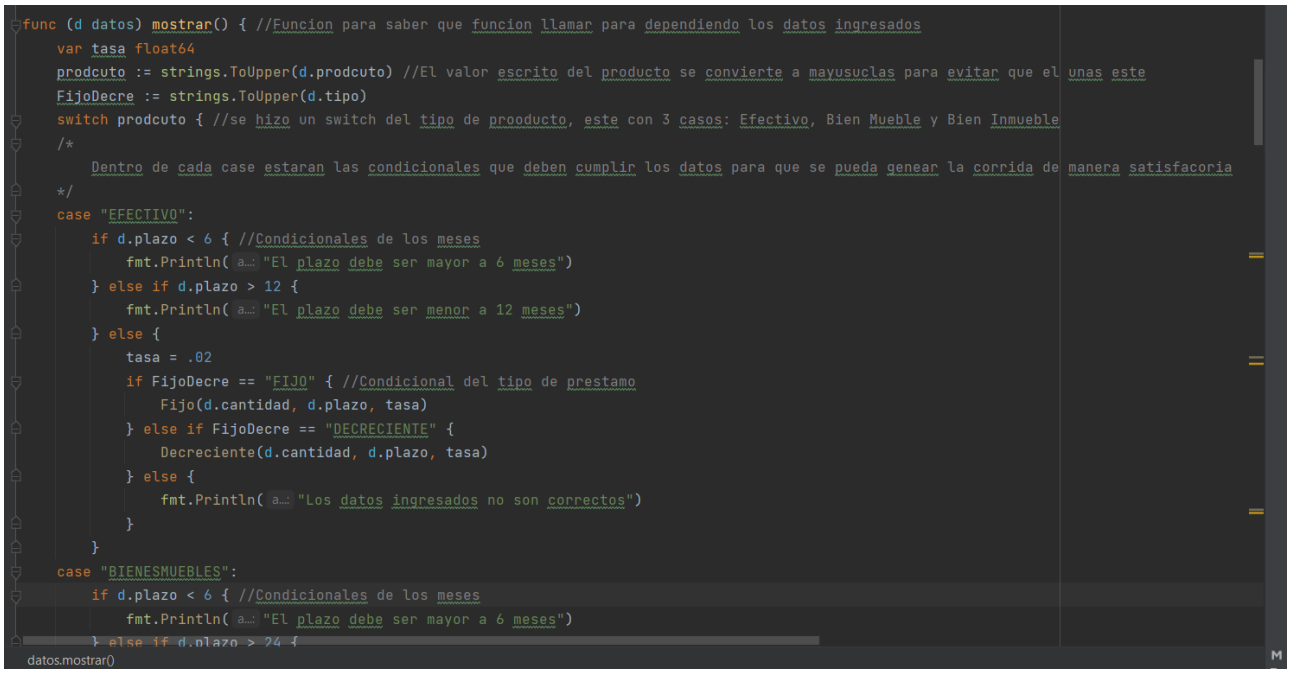
```
func main() {
    intCant, _ := strconv.ParseFloat(os.Args[2], bitSize: 64) //Como los datos que reciben son string, se convierten a flotantes
    intPlazo, _ := strconv.ParseFloat(os.Args[3], bitSize: 64)
    datoS := datos{ //Se reciben los datos del strcut mediante la terminal. Se toma desde el valor 1 y no desde el 0 por que ese es la direccion
        prodcuto: os.Args[1], //Estos datos no ocupan convertirse, ya recibe el parametro string
        cantidad: intCant,
        plazo:    intPlazo,
        tipo:     os.Args[4],
    }
    datoS.mostrar()
}
```

En el main se encuentra el código para recibir los datos mandados a través de la terminal. También para meterlos en la estructura de datos.

Algunos datos se tienen que convertir para poder ser utilizados.

Por ultimo se manda a llamar la función(metodo) para mostrar los datos.

Método mostrar()



```
func (d datos) mostrar() { //Funcion para saber que funcion llamar para dependiendo los datos ingresados
    var tasa float64
    prodcuto := strings.ToUpper(d.prodcuto) //El valor escrito del producto se convierte a mayusuclas para evitar que el unas este
    FijoDecre := strings.ToUpper(d.tipo)
    switch prodcuto { //se hizo un switch del tipo de prooducto, este con 3 casos: Efectivo, Bien Mueble y Bien Inmueble
    /*
        Dentro de cada case estaran las condicionales que deben cumplir los datos para que se pueda genear la corrida de manera satisfacoria
    */
    case "EFECTIVO":
        if d.plazo < 6 { //Condicionales de los meses
            fmt.Println( a...: "El plazo debe ser mayor a 6 meses")
        } else if d.plazo > 12 {
            fmt.Println( a...: "El plazo debe ser menor a 12 meses")
        } else {
            tasa = .02
            if FijoDecre == "FIJO" { //Condicional del tipo de prestamo
                Fijo(d.cantidad, d.plazo, tasa)
            } else if FijoDecre == "DECRECIENTE" {
                Decreciente(d.cantidad, d.plazo, tasa)
            } else {
                fmt.Println( a...: "Los datos ingresados no son correctos")
            }
        }
    case "BIENESMUEBLES":
        if d.plazo < 6 { //Condicionales de los meses
            fmt.Println( a...: "El plazo debe ser mayor a 6 meses")
        } else if d.plazo > 24 {
datos.mostrar()
```

Antes que nada, en go los métodos funcionan con tipos, en este caso el tipo es una estructura de datos llamada datos

```
type datos struct { //Estruct
    prodcuto string
    cantidad float64
    plazo    float64
    tipo     string
}
```

Esta estructura contiene un colección de campos de datos (dos cadenas y dos flotantes).

Continuando con el método, este cuenta con un switch que tiene como parámetro el producto ingresado, contiene un case para cada producto.

Adentro de cada case se encuentran las condicionales para que, si los datos son correctos, mande llamar la función del tipo de cuota de imprimación de corrida. En caso de que los datos estén incorrectos o no cuenten con las especificaciones necesarias, imprimirá un texto avisando al usuario.

Las funciones que manda a llamar una vez que se han verificado y aprobado las condiciones son 3, cada una recibe tres parámetros (El producto, Los meses o plazo en el que se hace el préstamo y la tasa de interés)

La primera función se llama Fijo()

```
func Fijo(cantidad float64, plazo float64, tasa float64) { //Funcion para las cuotas fijas
    saldoInicial := cantidad
    cuotaMensual := cantidad * ((tasa * math.Pow( x: (1+tasa), plazo)) / (math.Pow( x: (1+tasa), plazo) - 1)) //Formula para obtener la cuota mens
    amortización := cuotaMensual
    interes := cantidad * tasa
    saldoFinal := saldoInicial
    var sumaAmortizacion, sumaIntereses, sumaCuotaMensual float64 //Se crean las varables para poner el total de la sumas
    fmt.Println( a...: "                                                                                    ")
    fmt.Println( a...: "| No. Periodos | Amortización |  Interes  | Cuota Mensual |  Saldo final    |")
    fmt.Println( a...: "| -------------|--------------|-----------|---------------|---------------- |")
    for i := 0; i <= int(plazo); i++ { //Bucle para la corrida de los datos, se ejecutará las cantidad de meses del prestamo
        if i == 0 {
            fmt.Printf( format: "|      %3d     |       0      |     0     |        0      | %12.2f   |\n", i, saldoFinal) //En el periodo cero lo
        } else {
            fmt.Printf( format: "|      %3d     | %10.2f   |%10.2f | %10.2f    | %12.2f   |\n", i, amortización, interes, cuotaMensual, saldoFinal

        }
        saldoInicial = saldoFinal             //El saldo inicial siempre se inicia con el saldo final del periodo anterior
        interes = saldoFinal * tasa           //El interes siempre es por la deuda faltante
        amortización = cuotaMensual - interes //En el saldo fijo la amortizacion o pago capital es la cuota mensaul menos los intereses
        saldoFinal = saldoInicial - amortización
        sumaAmortizacion = sumaAmortizacion + amortización
        sumaIntereses = sumaIntereses + interes
        sumaCuotaMensual = sumaCuotaMensual + cuotaMensual
    }
```

Dentro de esta función se crean varias variables para los parámetros obtenidos, para las formulas y para al final poder tener las sumas totales de los datos.

Contiene un ciclo for que se recorre por la cantidad de meses del préstamo. Genera una tabla donde se muestran los periodos, los intereses, la amortización o pago a capital, la cuota mensual y el saldo total, estos datos se van actualizando conforme avanzan los periodos.

Ejemplo:

```
PS D:\6to Semestre\Análisis y Visualización de Datos (Walter)\examen> go run main.go
┌──────────────────────────────────────────────────────────────────────────────────┐
│ No. Periodos | Amortización |  Interes  | Cuota Mensual |  Saldo final   │
│ -------------|--------------|-----------|---------------|--------------- │
│        0     |       0      |     0     |        0      |       5000.00  │
│        1     |     582.55   |    100.00 |     682.55    |       4417.45  │
│        2     |     594.20   |     88.35 |     682.55    |       3823.25  │
│        3     |     606.08   |     76.47 |     682.55    |       3217.17  │
│        4     |     618.21   |     64.34 |     682.55    |       2598.96  │
│        5     |     630.57   |     51.98 |     682.55    |       1968.39  │
│        6     |     643.18   |     39.37 |     682.55    |       1325.21  │
│        7     |     656.04   |     26.50 |     682.55    |        669.17  │
│        8     |     669.17   |     13.38 |     682.55    |         -0.00  │
│ -------------|--------------|-----------|---------------|--------------- │
│      Total   |    5682.55   |    460.39 |    6142.94    |                │
PS D:\6to Semestre\Análisis y Visualización de Datos (Walter)\examen>
```

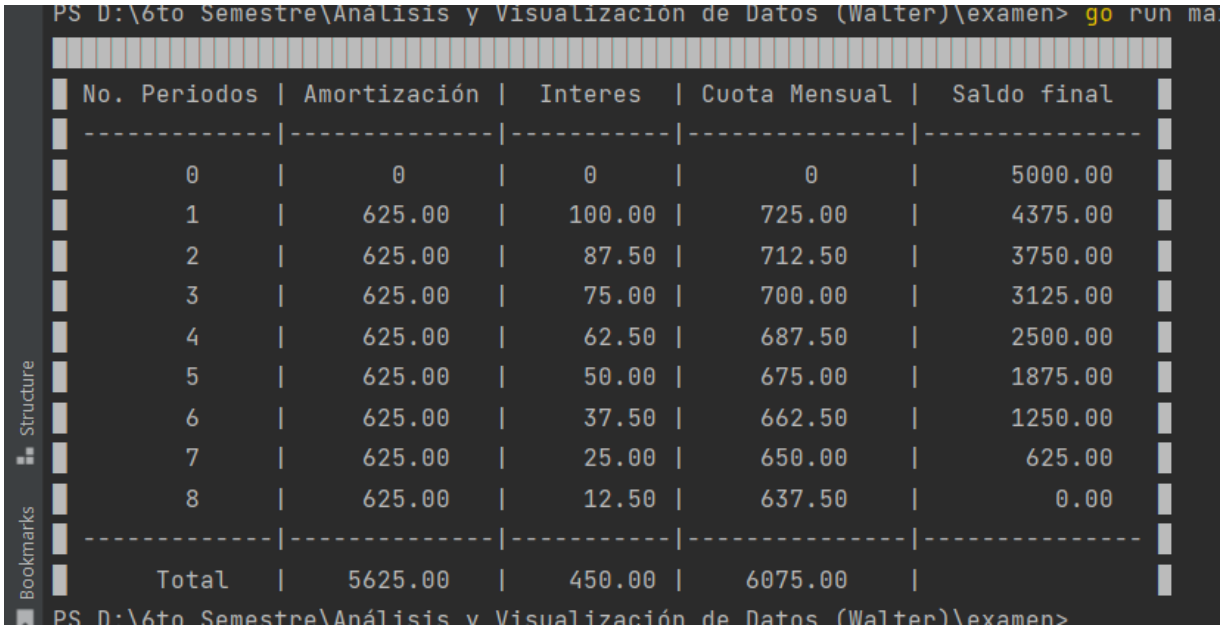
La segunda función se llama Decreciente()

```
func Decreciente(cantidad float64, plazo float64, tasa float64) { //Funcion para las cuotas decrecientes
    //Esta funcion se hizo diferente por que se le añade la funcionalidad de que d
    saldoInicial := cantidad
    cuotaMensual := (cantidad*tasa)/1 - math.Pow( x: (1+tasa), plazo) //Formula para
    amortización := cantidad / float64(plazo)
    interes := cantidad * tasa
    saldoFinal := saldoInicial
    var sumaAmortizacion, sumaIntereses, sumaCuotaMensual float64 //Se crean las varables para poner el total de la sumas
    fmt.Println( a...: "┌──────────────────────────────────────────────────────────────────────────────────┐")
    fmt.Println( a...: "│ No. Periodos | Amortización |  Interes  | Cuota Mensual |  Saldo final   │")
    fmt.Println( a...: "│ -------------|--------------|-----------|---------------|--------------- │")
    for i := 0; i <= int(plazo); i++ { //Bucle para la corrida de los datos, se ejecutará las cantidad de meses del prestamo
        if i == 0 {
            fmt.Printf( format: "│      %3d     |       0      |     0     |        0      | %12.2f   │\n", i, saldoFinal)
        } else {
            fmt.Printf( format: "│      %3d     | %10.2f   |%10.2f | %10.2f    | %12.2f   │\n", i, amortización, interes, cuotaMensual, saldoFinal
        }
        saldoInicial = saldoFinal
        interes = saldoFinal * tasa
        cuotaMensual = amortización + interes
        saldoFinal = saldoInicial - amortización
        sumaAmortizacion = sumaAmortizacion + amortización
        sumaIntereses = sumaIntereses + interes
        sumaCuotaMensual = sumaCuotaMensual + cuotaMensual
    }
    fmt.Println( a...: "│ -------------|--------------|-----------|---------------|--------------- │")
    fmt.Printf( format: "│      %s    | %10.2f   |%10.2f | %10.2f    |                │\n", a...: "Total", sumaAmortizacion, sumaIntereses, sumaCuotal
Fijo(cantidad float64, plazo float64, tasa float64) › cuotaMensual
```

Es casi idéntica a la función Fijo(), lo único que cambia es la formula para obtener la cuota mensual, la amortización o pago capital y el saldo final.

Fuera de eso contiene la misma estructura, el mismo ciclo con la misma funcionalidad.

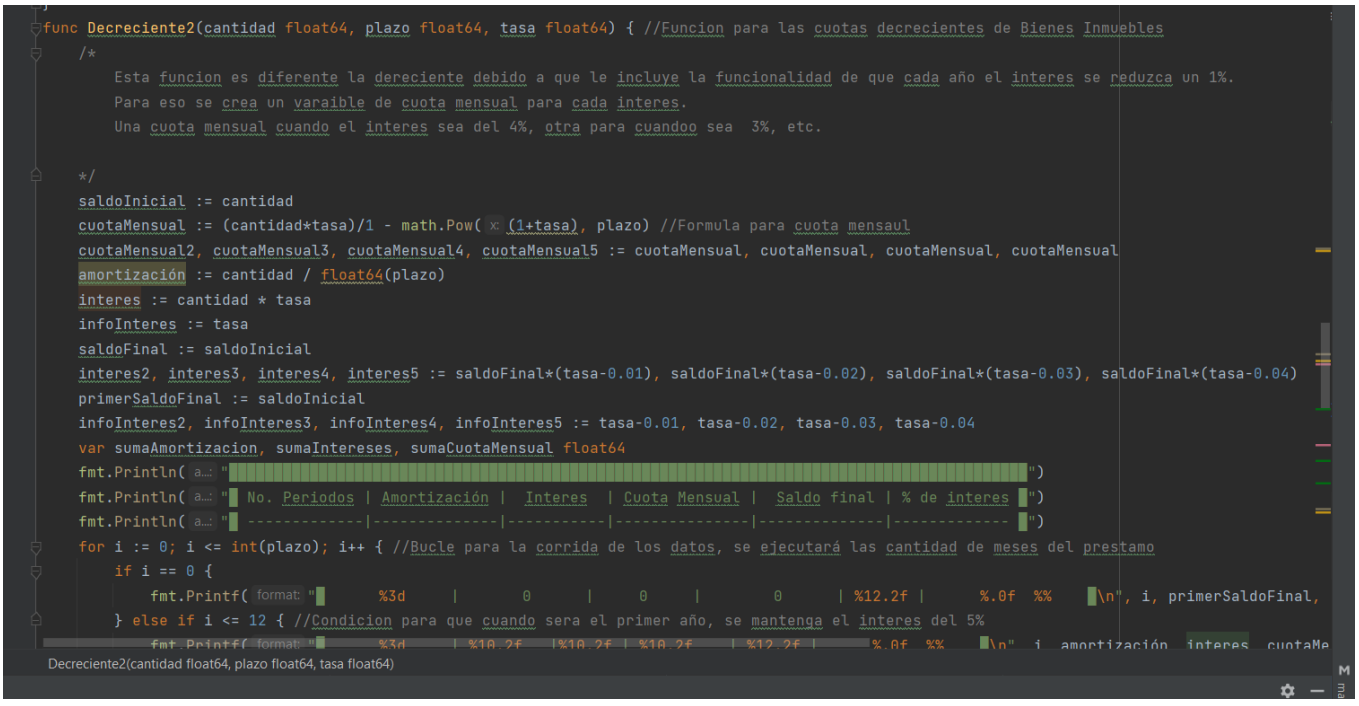
Ejemplo de corrida:



```
PS D:\6to Semestre\Análisis y Visualización de Datos (Walter)\examen> go run ma

| No. Periodos | Amortización |  Interes  | Cuota Mensual |  Saldo final  |
|--------------|--------------|-----------|---------------|---------------|
|       0      |       0      |     0     |       0       |      5000.00  |
|       1      |     625.00   |    100.00 |     725.00    |      4375.00  |
|       2      |     625.00   |     87.50 |     712.50    |      3750.00  |
|       3      |     625.00   |     75.00 |     700.00    |      3125.00  |
|       4      |     625.00   |     62.50 |     687.50    |      2500.00  |
|       5      |     625.00   |     50.00 |     675.00    |      1875.00  |
|       6      |     625.00   |     37.50 |     662.50    |      1250.00  |
|       7      |     625.00   |     25.00 |     650.00    |       625.00  |
|       8      |     625.00   |     12.50 |     637.50    |         0.00  |
|--------------|--------------|-----------|---------------|---------------|
|     Total    |    5625.00   |    450.00 |    6075.00    |               |
PS D:\6to Semestre\Análisis y Visualización de Datos (Walter)\examen>
```

Función Decreciente2()

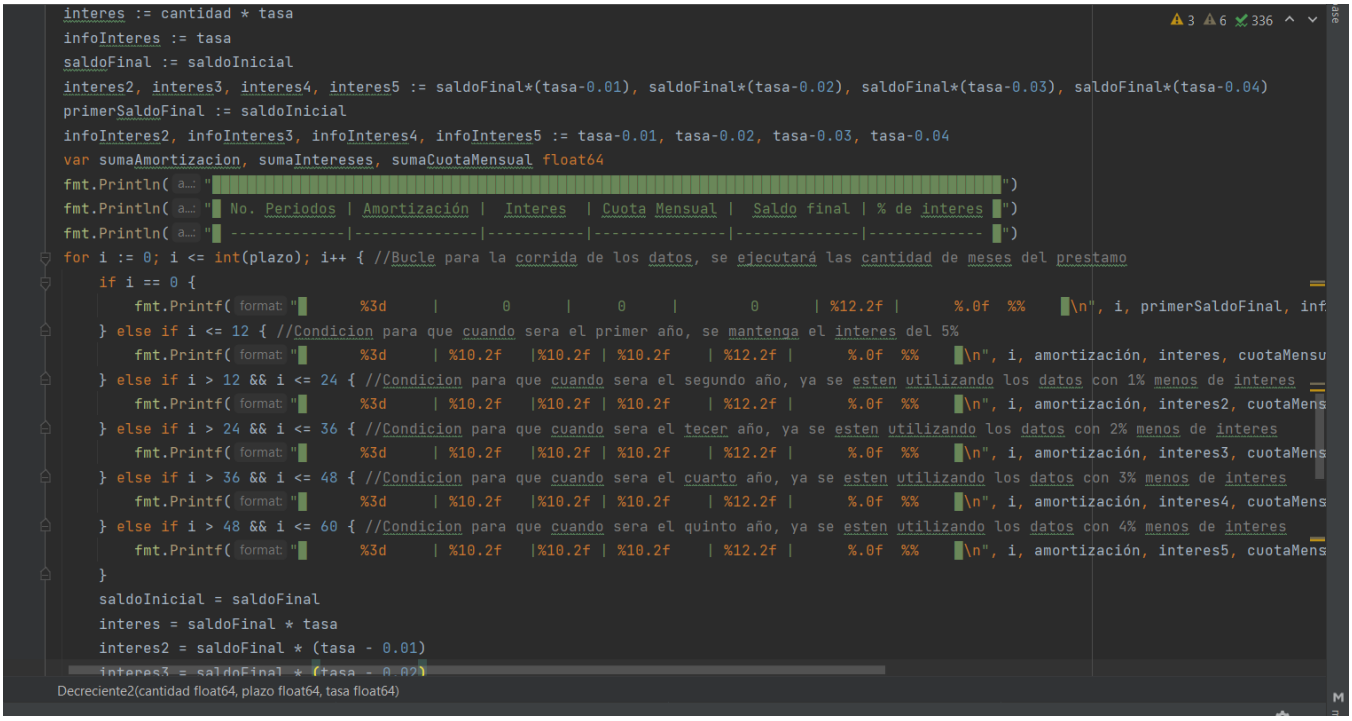


```
func Decreciente2(cantidad float64, plazo float64, tasa float64) { //Funcion para las cuotas decrecientes de Bienes Inmuebles
    /*
        Esta funcion es diferente la dereciente debido a que le incluye la funcionalidad de que cada año el interes se reduzca un 1%.
        Para eso se crea un varaible de cuota mensual para cada interes.
        Una cuota mensual cuando el interes sea del 4%, otra para cuandoo sea  3%, etc.

    */
    saldoInicial := cantidad
    cuotaMensual := (cantidad*tasa)/1 - math.Pow( x (1+tasa), plazo) //Formula para cuota mensaul
    cuotaMensual2, cuotaMensual3, cuotaMensual4, cuotaMensual5 := cuotaMensual, cuotaMensual, cuotaMensual, cuotaMensual
    amortización := cantidad / float64(plazo)
    interes := cantidad * tasa
    infoInteres := tasa
    saldoFinal := saldoInicial
    interes2, interes3, interes4, interes5 := saldoFinal*(tasa-0.01), saldoFinal*(tasa-0.02), saldoFinal*(tasa-0.03), saldoFinal*(tasa-0.04)
    primerSaldoFinal := saldoInicial
    infoInteres2, infoInteres3, infoInteres4, infoInteres5 := tasa-0.01, tasa-0.02, tasa-0.03, tasa-0.04
    var sumaAmortizacion, sumaIntereses, sumaCuotaMensual float64
    fmt.Println( a...: "                                                                                                    ")
    fmt.Println( a...: "| No. Periodos | Amortización |  Interes  | Cuota Mensual |  Saldo final | % de interes |")
    fmt.Println( a...: "|--------------|--------------|-----------|---------------|--------------|--------------|")
    for i := 0; i <= int(plazo); i++ { //Bucle para la corrida de los datos, se ejecutará las cantidad de meses del prestamo
        if i == 0 {
            fmt.Printf( format: "|      %3d     |       0      |     0     |       0       | %12.2f |      %.0f  %%    |\n", i, primerSaldoFinal,
        } else if i <= 12 { //Condicion para que cuando sera el primer año, se mantenga el interes del 5%
            fmt.Printf( format: "|      %3d     | %10.2f   | %10.2f | %10.2f    | %12.2f |      %.0f  %%    |\n", i, amortización, interes, cuotaMe
Decreciente2(cantidad float64, plazo float64, tasa float64)
```

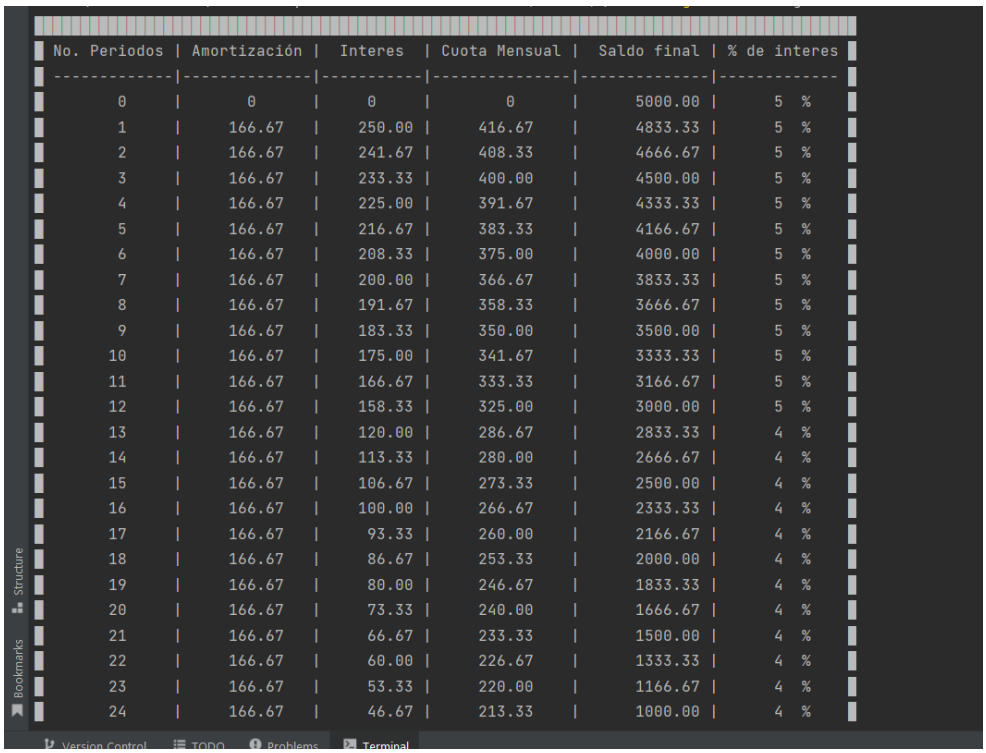
Esta función es un poco diferente a la decreciente debido a que le incluye la funcionalidad de que cada año el interés se reduzca un 1%. Para eso se crea un variable de cuota mensual para cada interés. Una cuota mensual cuando el interés sea del 4%, otra para cuando sea 3%, etc.

Dentro del ciclo se agregan condiciones para que dependiendo del año en el que vaya el periodo, se estén utilizando los datos con 1% menos de interés al años pasado.

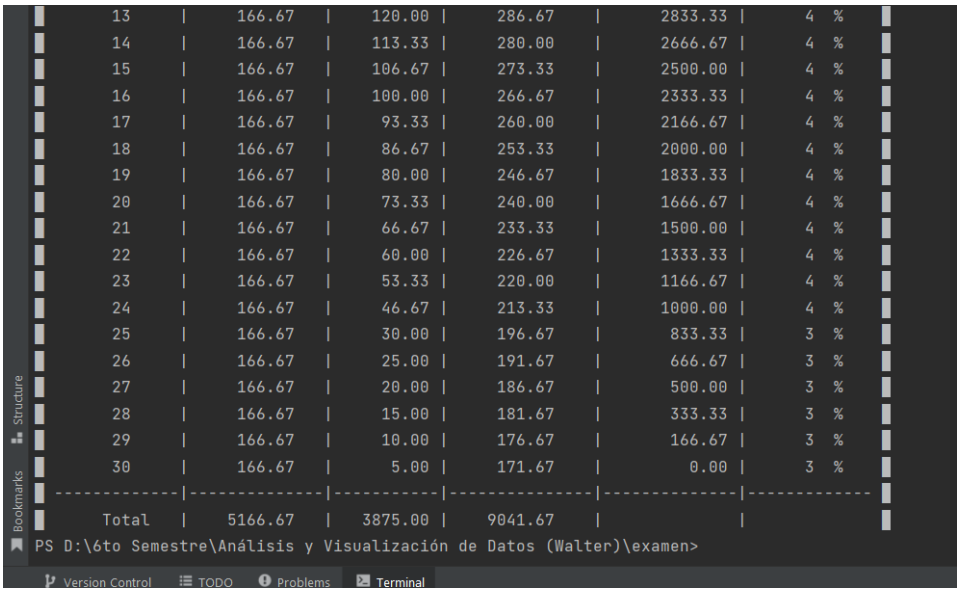


```
interes := cantidad * tasa
infoInteres := tasa
saldoFinal := saldoInicial
interes2, interes3, interes4, interes5 := saldoFinal*(tasa-0.01), saldoFinal*(tasa-0.02), saldoFinal*(tasa-0.03), saldoFinal*(tasa-0.04)
primerSaldoFinal := saldoInicial
infoInteres2, infoInteres3, infoInteres4, infoInteres5 := tasa-0.01, tasa-0.02, tasa-0.03, tasa-0.04
var sumaAmortizacion, sumaIntereses, sumaCuotaMensual float64
fmt.Println( a...: "                                                                                                    ")
fmt.Println( a...: "| No. Periodos | Amortización |  Interes  | Cuota Mensual |  Saldo final | % de interes |")
fmt.Println( a...: "| -------------|--------------|-----------|---------------|--------------|------------- |")
for i := 0; i <= int(plazo); i++ { //Bucle para la corrida de los datos, se ejecutará las cantidad de meses del prestamo
    if i == 0 {
        fmt.Printf( format: "|      %3d     |       0      |     0     |       0       | %12.2f |      %.0f  %%    |\n", i, primerSaldoFinal, inf
    } else if i <= 12 { //Condicion para que cuando sera el primer año, se mantenga el interes del 5%
        fmt.Printf( format: "|      %3d     | %10.2f   |%10.2f | %10.2f    | %12.2f |      %.0f  %%    |\n", i, amortización, interes, cuotaMensu
    } else if i > 12 && i <= 24 { //Condicion para que cuando sera el segundo año, ya se esten utilizando los datos con 1% menos de interes
        fmt.Printf( format: "|      %3d     | %10.2f   |%10.2f | %10.2f    | %12.2f |      %.0f  %%    |\n", i, amortización, interes2, cuotaMens
    } else if i > 24 && i <= 36 { //Condicion para que cuando sera el tecer año, ya se esten utilizando los datos con 2% menos de interes
        fmt.Printf( format: "|      %3d     | %10.2f   |%10.2f | %10.2f    | %12.2f |      %.0f  %%    |\n", i, amortización, interes3, cuotaMens
    } else if i > 36 && i <= 48 { //Condicion para que cuando sera el cuarto año, ya se esten utilizando los datos con 3% menos de interes
        fmt.Printf( format: "|      %3d     | %10.2f   |%10.2f | %10.2f    | %12.2f |      %.0f  %%    |\n", i, amortización, interes4, cuotaMens
    } else if i > 48 && i <= 60 { //Condicion para que cuando sera el quinto año, ya se esten utilizando los datos con 4% menos de interes
        fmt.Printf( format: "|      %3d     | %10.2f   |%10.2f | %10.2f    | %12.2f |      %.0f  %%    |\n", i, amortización, interes5, cuotaMens
    }
    saldoInicial = saldoFinal
    interes = saldoFinal * tasa
    interes2 = saldoFinal * (tasa - 0.01)
    interes3 = saldoFinal * (tasa - 0.02)
Decreciente2(cantidad float64, plazo float64, tasa float64)
```

Ejemplo de la impresión de datos:



No. Periodos	Amortización	Interes	Cuota Mensual	Saldo final	% de interes
0	0	0	0	5000.00	5 %
1	166.67	250.00	416.67	4833.33	5 %
2	166.67	241.67	408.33	4666.67	5 %
3	166.67	233.33	400.00	4500.00	5 %
4	166.67	225.00	391.67	4333.33	5 %
5	166.67	216.67	383.33	4166.67	5 %
6	166.67	208.33	375.00	4000.00	5 %
7	166.67	200.00	366.67	3833.33	5 %
8	166.67	191.67	358.33	3666.67	5 %
9	166.67	183.33	350.00	3500.00	5 %
10	166.67	175.00	341.67	3333.33	5 %
11	166.67	166.67	333.33	3166.67	5 %
12	166.67	158.33	325.00	3000.00	5 %
13	166.67	120.00	286.67	2833.33	4 %
14	166.67	113.33	280.00	2666.67	4 %
15	166.67	106.67	273.33	2500.00	4 %
16	166.67	100.00	266.67	2333.33	4 %
17	166.67	93.33	260.00	2166.67	4 %
18	166.67	86.67	253.33	2000.00	4 %
19	166.67	80.00	246.67	1833.33	4 %
20	166.67	73.33	240.00	1666.67	4 %
21	166.67	66.67	233.33	1500.00	4 %
22	166.67	60.00	226.67	1333.33	4 %
23	166.67	53.33	220.00	1166.67	4 %
24	166.67	46.67	213.33	1000.00	4 %



13	166.67	120.00	286.67	2833.33	4 %
14	166.67	113.33	280.00	2666.67	4 %
15	166.67	106.67	273.33	2500.00	4 %
16	166.67	100.00	266.67	2333.33	4 %
17	166.67	93.33	260.00	2166.67	4 %
18	166.67	86.67	253.33	2000.00	4 %
19	166.67	80.00	246.67	1833.33	4 %
20	166.67	73.33	240.00	1666.67	4 %
21	166.67	66.67	233.33	1500.00	4 %
22	166.67	60.00	226.67	1333.33	4 %
23	166.67	53.33	220.00	1166.67	4 %
24	166.67	46.67	213.33	1000.00	4 %
25	166.67	30.00	196.67	833.33	3 %
26	166.67	25.00	191.67	666.67	3 %
27	166.67	20.00	186.67	500.00	3 %
28	166.67	15.00	181.67	333.33	3 %
29	166.67	10.00	176.67	166.67	3 %
30	166.67	5.00	171.67	0.00	3 %
Total	5166.67	3875.00	9041.67		

Código:

```
package main

import (
   "fmt"
   "math"
   "os"
   "strconv"
   "strings"
)

type datos struct { //Estructura de los datos que van a recibir
   prodcuto string
   cantidad float64
   plazo    float64
   tipo     string
}

func (d datos) mostrar() { //Funcion para saber que funcion llamar para
dependiendo los datos ingresados
   var tasa float64
   prodcuto := strings.ToUpper(d.prodcuto) //El valor escrito del
producto se convierte a mayusuclas para evitar que el unas este
   FijoDecre := strings.ToUpper(d.tipo)
   switch prodcuto { //se hizo un switch del tipo de prooducto, este con
3 casos: Efectivo, Bien Mueble y Bien Inmueble
   /*
      Dentro de cada case estaran las condicionales que deben cumplir los
datos para que se pueda genear la corrida de manera satisfacoria
   */
   case "EFECTIVO":
      if d.plazo < 6 { //Condicionales de los meses
         fmt.Println("El plazo debe ser mayor a 6 meses")
      } else if d.plazo > 12 {
         fmt.Println("El plazo debe ser menor a 12 meses")
      } else {
         tasa = .02
```

```
         if FijoDecre == "FIJO" { //Condicional del tipo de prestamo
            Fijo(d.cantidad, d.plazo, tasa)
         } else if FijoDecre == "DECRECIENTE" {
            Decreciente(d.cantidad, d.plazo, tasa)
         } else {
            fmt.Println("Los datos ingresados no son correctos")
         }
      }
   case "BIENESMUEBLES":
      if d.plazo < 6 { //Condicionales de los meses
         fmt.Println("El plazo debe ser mayor a 6 meses")
      } else if d.plazo > 24 {
         fmt.Println("El plazo debe ser menor a 24 meses")
      } else {
         tasa = .025
         if FijoDecre == "FIJO" { //Condicional del tipo de prestamo
            Fijo(d.cantidad, d.plazo, tasa)
         } else if FijoDecre == "DECRECIENTE" {
            Decreciente(d.cantidad, d.plazo, tasa)
         } else {
            fmt.Println("Los datos ingresados no son correctos")
         }
      }
   case "BIENESINMUEBLES":
      if d.plazo < 12 { //Condicionales de los meses
         fmt.Println("El plazo debe ser mayor a 12 meses")
      } else if d.plazo > 60 {
         fmt.Println("El plazo debe ser menor a 60 meses")
      } else {
         tasa = .05
         if FijoDecre == "FIJO" { //Condicional del tipo de prestamo
            Fijo(d.cantidad, d.plazo, tasa)
         } else if FijoDecre == "DECRECIENTE" {
            Decreciente2(d.cantidad, d.plazo, tasa)
         } else {
            fmt.Println("Los datos ingresados no son correctos")
         }
      }
   default:
      fmt.Println("Los datos ingresados no son correctos")
   }
}
func Decreciente(cantidad float64, plazo float64, tasa float64) {
//Funcion para las cuotas decrecientes
   //Esta funcion se hizo diferente por que se le añade la funcionalidad
de que d
   saldoInicial := cantidad
   cuotaMensual := (cantidad*tasa)/1 - math.Pow((1+tasa), plazo)
//Formula para
   amortización := cantidad / float64(plazo)
   interes := cantidad * tasa
   saldoFinal := saldoInicial
   var sumaAmortizacion, sumaIntereses, sumaCuotaMensual float64 //Se
crean las varables para poner el total de la sumas

fmt.Println("                                                              
              ")
```

```
   fmt.Println("| No. Periodos | Amortización |  Interes  | Cuota Mensual
|  Saldo final    |")
   fmt.Println("|--------------|--------------|-----------|--------------
-|---------------|")
   for i := 0; i <= int(plazo); i++ { //Bucle para la corrida de los
datos, se ejecutará las cantidad de meses del prestamo
      if i == 0 {
         fmt.Printf("|       %3d      |       0      |     0     |         0
|  %12.2f   |\n", i, saldoFinal)
      } else {
         fmt.Printf("|       %3d      | %10.2f   |%10.2f | %10.2f    |
%12.2f   |\n", i, amortización, interes, cuotaMensual, saldoFinal)
      }
      saldoInicial = saldoFinal
      interes = saldoFinal * tasa
      cuotaMensual = amortización + interes
      saldoFinal = saldoInicial - amortización
      sumaAmortizacion = sumaAmortizacion + amortización
      sumaIntereses = sumaIntereses + interes
      sumaCuotaMensual = sumaCuotaMensual + cuotaMensual
   }
   fmt.Println("|--------------|--------------|-----------|--------------
-|---------------|")
   fmt.Printf("|       %s    | %10.2f   |%10.2f | %10.2f    |
\n", "Total", sumaAmortizacion, sumaIntereses, sumaCuotaMensual)
}
func Decreciente2(cantidad float64, plazo float64, tasa float64) {
//Funcion para las cuotas decrecientes de Bienes Inmuebles
   /*
      Esta funcion es diferente la decreciente debido a que le incluye la
funcionalidad de que cada año el interes se reduzca un 1%.
      Para eso se crea un varaible de cuota mensual para cada interes.
      Una cuota mensual cuando el interes sea del 4%, otra para cuando
sea  3%, etc.
   */
   saldoInicial := cantidad
   cuotaMensual := (cantidad*tasa)/1 - math.Pow((1+tasa), plazo)
//Formula para cuota mensaul
   cuotaMensual2, cuotaMensual3, cuotaMensual4, cuotaMensual5 :=
cuotaMensual, cuotaMensual, cuotaMensual, cuotaMensual
   amortización := cantidad / float64(plazo)
   interes := cantidad * tasa
   infoInteres := tasa
   saldoFinal := saldoInicial
   interes2, interes3, interes4, interes5 := saldoFinal*(tasa-0.01),
saldoFinal*(tasa-0.02), saldoFinal*(tasa-0.03), saldoFinal*(tasa-0.04)
   primerSaldoFinal := saldoInicial
   infoInteres2, infoInteres3, infoInteres4, infoInteres5 := tasa-0.01,
tasa-0.02, tasa-0.03, tasa-0.04
   var sumaAmortizacion, sumaIntereses, sumaCuotaMensual float64

fmt.Println("                                                                
                                ")
   fmt.Println("| No. Periodos | Amortización |  Interes  | Cuota Mensual
|  Saldo final | % de interes |")
   fmt.Println("|--------------|--------------|-----------|--------------
-|---------------|-------------- |")
```

```
   for i := 0; i <= int(plazo); i++ { //Bucle para la corrida de los
datos, se ejecutará las cantidad de meses del prestamo
      if i == 0 {
         fmt.Printf("▒       %3d      |        0       |     0     |         0
| %12.2f |       %.0f  %%    ▒\n", i, primerSaldoFinal, infoInteres*100)
      } else if i <= 12 { //Condicion para que cuando sera el primer año,
se mantenga el interes del 5%
         fmt.Printf("▒       %3d      | %10.2f   |%10.2f | %10.2f    |
%12.2f |       %.0f  %%    ▒\n", i, amortización, interes, cuotaMensual,
saldoFinal, infoInteres*100)
      } else if i > 12 && i <= 24 { //Condicion para que cuando sera el
segundo año, ya se esten utilizando los datos con 1% menos de interes
         fmt.Printf("▒       %3d      | %10.2f   |%10.2f | %10.2f    |
%12.2f |       %.0f  %%    ▒\n", i, amortización, interes2, cuotaMensual2,
saldoFinal, infoInteres2*100)
      } else if i > 24 && i <= 36 { //Condicion para que cuando sera el
tecer año, ya se esten utilizando los datos con 2% menos de interes
         fmt.Printf("▒       %3d      | %10.2f   |%10.2f | %10.2f    |
%12.2f |       %.0f  %%    ▒\n", i, amortización, interes3, cuotaMensual3,
saldoFinal, infoInteres3*100)
      } else if i > 36 && i <= 48 { //Condicion para que cuando sera el
cuarto año, ya se esten utilizando los datos con 3% menos de interes
         fmt.Printf("▒       %3d      | %10.2f   |%10.2f | %10.2f    |
%12.2f |       %.0f  %%    ▒\n", i, amortización, interes4, cuotaMensual4,
saldoFinal, infoInteres4*100)
      } else if i > 48 && i <= 60 { //Condicion para que cuando sera el
quinto año, ya se esten utilizando los datos con 4% menos de interes
         fmt.Printf("▒       %3d      | %10.2f   |%10.2f | %10.2f    |
%12.2f |       %.0f  %%    ▒\n", i, amortización, interes5, cuotaMensual5,
saldoFinal, infoInteres5*100)
      }
      saldoInicial = saldoFinal
      interes = saldoFinal * tasa
      interes2 = saldoFinal * (tasa - 0.01)
      interes3 = saldoFinal * (tasa - 0.02)
      interes4 = saldoFinal * (tasa - 0.03)
      interes5 = saldoFinal * (tasa - 0.04)
      cuotaMensual = amortización + interes
      cuotaMensual2 = amortización + interes2
      cuotaMensual3 = amortización + interes3
      cuotaMensual4 = amortización + interes4
      cuotaMensual5 = amortización + interes5
      saldoFinal = saldoInicial - amortización
      sumaAmortizacion = sumaAmortizacion + amortización
      sumaIntereses = sumaIntereses + interes
      sumaCuotaMensual = sumaCuotaMensual + cuotaMensual
   }
   fmt.Println("▒ -------------|--------------|-----------|--------------
-|--------------|------------- ▒")
   fmt.Printf("▒       %s    | %10.2f   |%10.2f | %10.2f    |
|              ▒\n", "Total", sumaAmortizacion, sumaIntereses,
sumaCuotaMensual)
}
func Fijo(cantidad float64, plazo float64, tasa float64) { //Funcion para
las cuotas fijas
   saldoInicial := cantidad
   cuotaMensual := cantidad * ((tasa * math.Pow((1+tasa), plazo)) /
```

```
(math.Pow((1+tasa), plazo) - 1)) //Formula para obtener la cuota mensual
fija
   amortización := cuotaMensual
   interes := cantidad * tasa
   saldoFinal := saldoInicial
   var sumaAmortizacion, sumaIntereses, sumaCuotaMensual float64 //Se
crean las varables para poner el total de la sumas

fmt.Println("┌──────────────────────────────────────────────────────────────
──────────────┐")
   fmt.Println("│ No. Periodos | Amortización |  Interes  | Cuota Mensual
|  Saldo final   │")
   fmt.Println("│ -------------|--------------|-----------|--------------
-|---------------- │")
   for i := 0; i <= int(plazo); i++ { //Bucle para la corrida de los
datos, se ejecutará las cantidad de meses del prestamo
      if i == 0 {
         fmt.Printf("│       %3d     |       0      |     0     |        0
| %12.2f    │\n", i, saldoFinal) //En el periodo cero lo unico que imprime
es el saldo final que es con el se empiza la corrida
      } else {
         fmt.Printf("│       %3d     | %10.2f   |%10.2f | %10.2f    |
%12.2f    │\n", i, amortización, interes, cuotaMensual, saldoFinal)

      }
      saldoInicial = saldoFinal             //El saldo inicial siempre se
inicia con el saldo final del periodo anterior
      interes = saldoFinal * tasa           //El interes siempre es por
la deuda faltante
      amortización = cuotaMensual - interes //En el saldo fijo la
amortizacion o pago capital es la cuota mensaul menos los intereses
      saldoFinal = saldoInicial - amortización
      sumaAmortizacion = sumaAmortizacion + amortización
      sumaIntereses = sumaIntereses + interes
      sumaCuotaMensual = sumaCuotaMensual + cuotaMensual
   }
   fmt.Println("│ -------------|--------------|-----------|--------------
-|---------------- │")
   fmt.Printf("│       %s    | %10.2f   |%10.2f | %10.2f    |
│\n", "Total", sumaAmortizacion, sumaIntereses, sumaCuotaMensual)
}
func main() {
   intCant, _ := strconv.ParseFloat(os.Args[2], 64) //Como los datos que
reciben son string, se convierten a flotantes
   intPlazo, _ := strconv.ParseFloat(os.Args[3], 64)
   datoS := datos{ //Se reciben los datos del strcut mediante la
terminal. Se toma desde el valor 1 y no desde el 0 por que ese es la
direccion en donde se guarda.
      prodcuto: os.Args[1], //Estos datos no ocupan convertirse, ya
recibe el parametro string
      cantidad: intCant,
      plazo:    intPlazo,
      tipo:     os.Args[4],
   }
   datoS.mostrar()
}
```