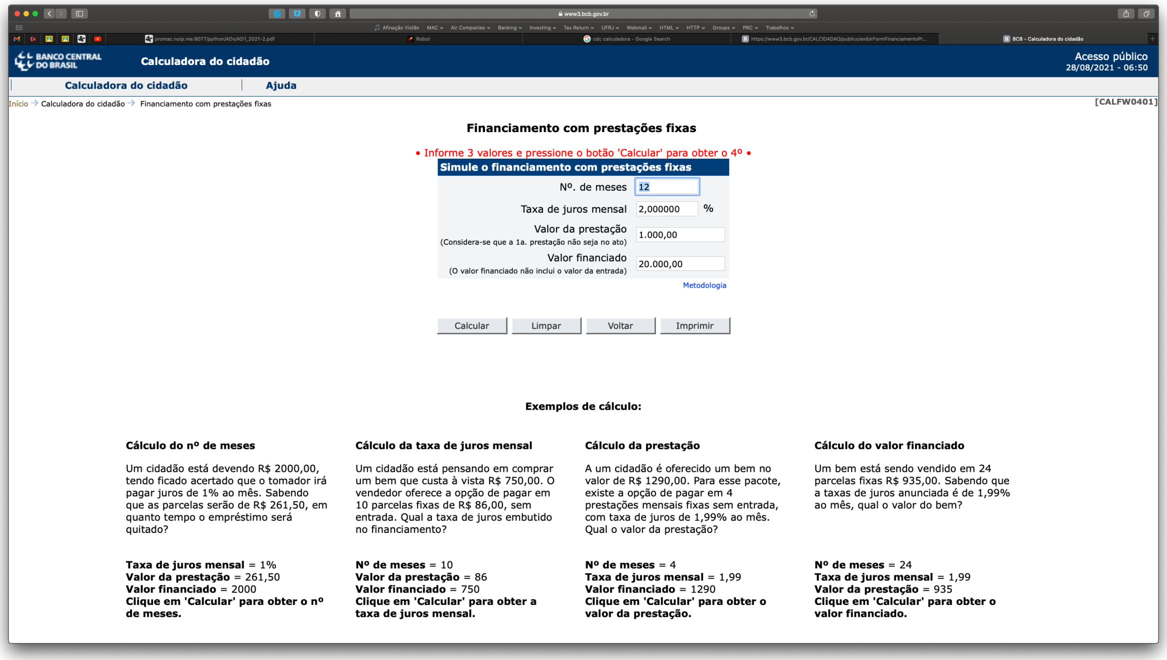
Disciplina: Desenvolvimento Web I
AD2 2^o semestre de 2021.
Professor: Paulo Roma

AD2: Desconto Racional por Dentro

1 Objetivo

O objetivo da AD2 é complementar as tarefas que ficaram faltando na AD1:

- O projeto proposto na AD1 é extremamente prático e atual. Ele versa sobre juros compostos e diversos bancos oferecem algo similar, via uma interface gráfica acessada com a utilização de um navegador qualquer. Por exemplo, considere-se a calculadora do cidadão do Banco Central do Brasil[1]. Embora ela tenha sido implementada pelo BC, ela é mal projetada, visualmente feia, e se alguém tentar preencher valores da tabela, provavelmente obterá um erro dizendo algo como: "• Informe 3 valores e pressione o botão 'Calcular' para obter o 4º •", conforme pode ser visto na Figura 1.



Financiamento com prestações fixas

• Informe 3 valores e pressione o botão 'Calcular' para obter o 4º •

Simule o financiamento com prestações fixas

Nº. de meses	12
Taxa de juros mensal	2,000000 %
Valor da prestação (Considera-se que a 1a. prestação não seja no ato)	1.000,00
Valor financiado (O valor financiado não inclui o valor da entrada)	20.000,00

Exemplos de cálculo:

Cálculo do nº de meses

Um cidadão está devendo R$ 2000,00, tendo ficado acertado que o tomador irá pagar juros de 1% ao mês. Sabendo que as parcelas serão de R$ 261,50, em quanto tempo o empréstimo será quitado?

Taxa de juros mensal = 1%
Valor da prestação = 261,50
Valor financiado = 2000
Clique em 'Calcular' para obter o nº de meses.

Cálculo da taxa de juros mensal

Um cidadão está pensando em comprar um bem que custa à vista R$ 750,00. O vendedor oferece a opção de pagar em 10 parcelas fixas de R$ 86,00, sem entrada. Qual a taxa de juros embutido no financiamento?

Nº de meses = 10
Valor da prestação = 86
Valor financiado = 750
Clique em 'Calcular' para obter a taxa de juros mensal.

Cálculo da prestação

A um cidadão é oferecido um bem no valor de R$ 1290,00. Para esse pacote, existe a opção de pagar em 4 prestações mensais fixas sem entrada, com taxa de juros de 1,99% ao mês. Qual o valor da prestação?

Nº de meses = 4
Taxa de juros mensal = 1,99
Valor financiado = 1290
Clique em 'Calcular' para obter o valor da prestação.

Cálculo do valor financiado

Um bem está sendo vendido em 24 parcelas fixas R$ 935,00. Sabendo que a taxas de juros anunciada é de 1,99% ao mês, qual o valor do bem?

Nº de meses = 24
Taxa de juros mensal = 1,99
Valor da prestação = 935
Clique em 'Calcular' para obter o valor financiado.

Figura 1: Calculadora do Cidadão.

Nesse caso, restam três opções:

[1] `https://www3.bcb.gov.br/CALCIDADAO/publico/exibirFormFinanciamentoPrestacoesFixas.do?method=exibirFormFinanciamentoPrestacoesFixas`

  - tentar adivinhar como o programa funciona, ou
  - procurar uma alternativa diferente de alguma outra instituição financeira, ou
  - implementar a sua própria calculadora.

  Portanto, como nosso curso é sobre interfaces gráficas, vamos analisar três maneiras diferentes de implementar uma interface funcional para o problema em questão.

1: Uma interface textual exibida diretamente numa janela do seu terminal.

2: Utilizando componentes de interface apropriados.

  - Normalmente, usuários não estão acostumados a usar terminais que executam um shell qualquer, como bash ou tcsh.
  - Portanto, é mais indicado criar a calculadora usando componentes de interface adequados fornecidos pelo tkinter[2]. Não existe uma forma fixa. Tente ser criativo e implemente a interface mais apropriada.

3: Através de uma interface gráfica implementada com HTML/CSS e que roda em um navegador qualquer[3] [4].

Nessa AD2, você deverá implementar as duas primeiras opções, apenas. A terceira opção foi abordada de alguma forma na AD1, mas se você quiser tentar escrever uma interface HTML/CSS diferente da nossa, sinta-se à vontade...

[2]`https://docs.python.org/3/library/tkinter.html`
[3]`http://orion.lcg.ufrj.br/python/html/cdc.html`
[4]`http://orion.lcg.ufrj.br/python/html/cdi.html`

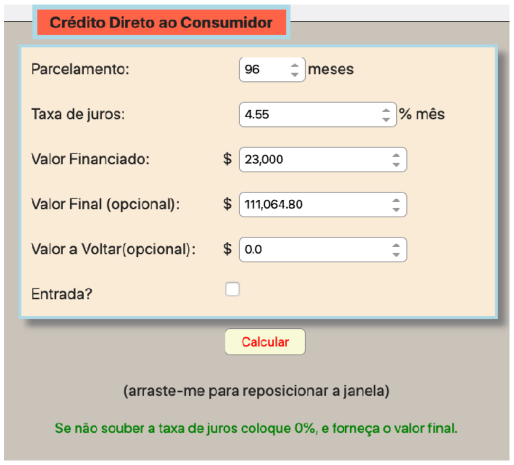
2 Diretivas Gerais

Para ajudá-lo a implementar uma interface textual, fornecemos o código básico para manter a interação com o usuário.

```
## Programa principal para testes.
#
#  @param argv command line arguments:
#  - h help
#  - n número de parcelas.
#  - t taxa mensal.
#  - x valor da compra a prazo.
#  - y valor da compra à vista.
#  - e indica uma entrada.
#  - v verbose mode
#
#  Usage:
#  - _02f_rational.py -n10 -t1 -x500 -y450 -e
#  - _02f_rational.py -n18 -t0 -x3297.60 -y1999
#  - _02f_rational.py -n10 -t0 -x1190 -y1094.80
#  - _02f_rational.py -n 88 -t 4.55 -x 111064.80 -y 23000
#  - _02f_rational.py -n 96 -t 0 -x 134788.8 -y 63816.24
#  - _02f_rational.py -n 4 -t 3.0 -x 1076.11  -y 1000
#  - _02f_rational.py --parcelas=88 --taxa=4.55 --valorP=111064.80 --valorV=23000 -v
#  - _02f_rational.py --help
#
#  @see https://mkaz.blog/code/python-string-format-cookbook/
#  @see https://www.w3schools.com/python/ref_string_format.asp
#  @see https://pyformat.info
#
def main(argv=None):
    if argv is None:
        argv = sys.argv

    # number of payments.
    np = 0
    # interest rate
    t = 0
    # initial price
    pv = 0
    # final price
    pp = 0
    # debugging state.
    debug = False
    # holds the existence of a down payment.
    setDownPayment(False)

    try:
        try:
            # options that require an argument should be followed by a colon (:)
            # long options, which require an argument, should be followed by an equal sign (=)
            opts, args = getopt.getopt(argv[1:], "hn:t:x:y:ev",
                ["help", "parcelas=", "taxa=", "valorP=", "valorV=", "entrada", "verbose"])
        except getopt.GetoptError as msg:
            raise ValueError(str(msg))
        # opts is an option list of pairs [(option1, argument1), (option2, argument2)]
        # args is the list of program arguments left after the option list was stripped
        # for instance, "move.py -h --help 1 2", sets opts and args to:
        # [('-h', ''), ('--help', '')] ['1', '2']
        for opt, arg in opts:  # something such as [('-h', '')] or [('--help', '')]
            if opt in ("-h", "--help"):
                print("Usage %s -n <nº parcelas> -t <taxa> -x <valor a prazo> " \
                               "-y <valor à vista> -e -v" % argv[0])
                return 1
            elif opt in ("-n", "--parcelas"):
                np = int(arg)
            elif opt in ("-t", "--taxa"):
                t = float(arg)/100.0
            elif opt in ("-x", "--valorP"):
                pp = float(arg)
            elif opt in ("-y", "--valorV"):
                pv = float(arg)
            elif opt in ("-v", "--verbose"):
                debug = True
            elif opt in ("-e", "--entrada"):
                setDownPayment()
    except ValueError as err:
        print(str(err) + "\nFor help, type: %s --help" % argv[0])
        return 2

    while np <= 0 or pv <= 0:
        try:
            np = int(input("Forneça o número de parcelas: "))
            t  = float(input("Forneça a taxa de juros: "))/100.0
            pp = float(input("Forneça o preço a prazo: "))
            pv = float(input("Forneça o preço à vista: "))
        except (EOFError, SyntaxError, ValueError, NameError, KeyboardInterrupt) as err:
            setDownPayment()
            rational_discount(10, 0.01, 500, 450, debug)
            sys.exit(err)

    if t > 0:
        if pp <= 0:
            (_, pp) = futureValue(pv, np, t)
        rational_discount(np, t, pp, pv, debug)
    else:
        t, ni = getInterest(pp, pv, np)
        print("Taxa = %.4f %% - %d iterações" % (t, ni))
        t *= 0.01
        print()
        rational_discount(np, t, pp, pv, debug)

    print()
    cf  = CF(t, np)
    pmt = pv*cf
    if getDownPayment():
        pmt /= (1+t)
        np -= 1    # uma prestação a menos
        pv -= pmt  # preço à vista menos a entrada
        print("Valor financiado = %.2f - %.2f = %.2f" % (pv+pmt, pmt, pv))

    print("Coeficiente de Financiamento: %f" % cf)
    print("Prestação: $%.2f" % pmt)

    # Tabela Price
    if debug:
        printTable(priceTable(np, pv, t, pmt))

if __name__ == "__main__":
    sys.exit(main())
```

Código 1: _02f_rational.py

O código 1 permite executar a calculadora no modo texto, sem o uso de qualquer tipo de interface gráfica, como pode ser visto abaixo, no código 2, em substituição a interface HTML/CSS da Figura 2.



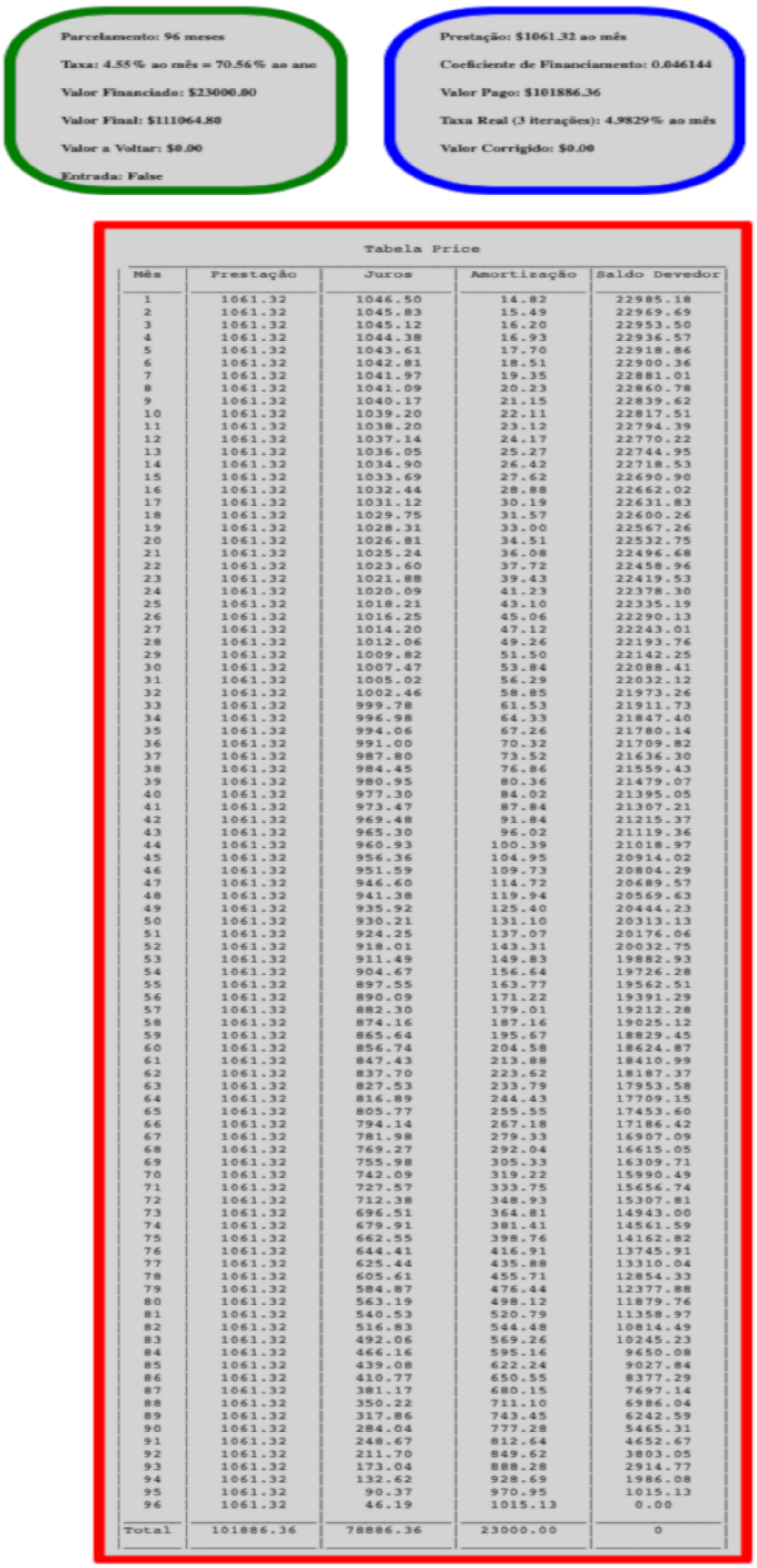
Crédito Direto ao Consumidor

Parcelamento:	96	meses
Taxa de juros:	4.55	% mês
Valor Financiado:	$ 23,000	
Valor Final (opcional):	$ 111,064.80	
Valor a Voltar(opcional):	$ 0.0	
Entrada?	☐	

(arraste-me para reposicionar a janela)

Se não souber a taxa de juros coloque 0%, e forneça o valor final.

Figura 2: Calculadora CDC - entrada.



Parcelamento: 96 meses

Taxa: 4.55% ao mês = 70.56% ao ano

Valor Financiado: $23000.00

Valor Final: $111064.80

Valor a Voltar: $0.00

Entrada: False

Prestação: $1061.32 ao mês

Coeficiente de Financiamento: 0.046144

Valor Pago: $101886.36

Taxa Real (3 iterações): 4.9829% ao mês

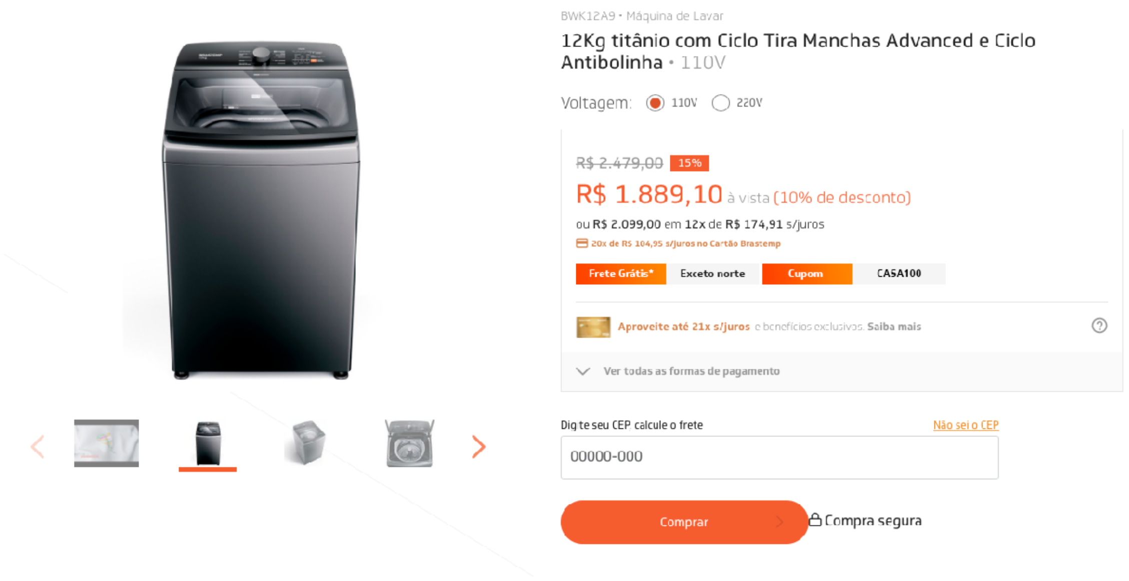
Valor Corrigido: $0.00

```
                              Tabela Price
 | Mês  |  Prestação  |    Juros    | Amortização |Saldo Devedor|
 |______|_____________|_____________|_____________|_____________|
 |   1  |   1061.32   |   1046.50   |    14.82    |  22985.18   |
 |   2  |   1061.32   |   1045.83   |    15.49    |  22969.69   |
 |   3  |   1061.32   |   1045.12   |    16.20    |  22953.50   |
 |   4  |   1061.32   |   1044.38   |    16.93    |  22936.57   |
 |   5  |   1061.32   |   1043.61   |    17.70    |  22918.86   |
 |   6  |   1061.32   |   1042.81   |    18.51    |  22900.36   |
 |   7  |   1061.32   |   1041.97   |    19.35    |  22881.01   |
 |   8  |   1061.32   |   1041.09   |    20.23    |  22860.78   |
 |   9  |   1061.32   |   1040.17   |    21.15    |  22839.62   |
 |  10  |   1061.32   |   1039.20   |    22.11    |  22817.51   |
 |  11  |   1061.32   |   1038.20   |    23.12    |  22794.39   |
 |  12  |   1061.32   |   1037.14   |    24.17    |  22770.22   |
 |  13  |   1061.32   |   1036.05   |    25.27    |  22744.95   |
 |  14  |   1061.32   |   1034.90   |    26.42    |  22718.53   |
 |  15  |   1061.32   |   1033.69   |    27.62    |  22690.90   |
 |  16  |   1061.32   |   1032.44   |    28.88    |  22662.02   |
 |  17  |   1061.32   |   1031.12   |    30.19    |  22631.83   |
 |  18  |   1061.32   |   1029.75   |    31.57    |  22600.26   |
 |  19  |   1061.32   |   1028.31   |    33.00    |  22567.26   |
 |  20  |   1061.32   |   1026.81   |    34.51    |  22532.75   |
 |  21  |   1061.32   |   1025.24   |    36.08    |  22496.68   |
 |  22  |   1061.32   |   1023.60   |    37.72    |  22458.96   |
 |  23  |   1061.32   |   1021.88   |    39.43    |  22419.53   |
 |  24  |   1061.32   |   1020.09   |    41.23    |  22378.30   |
 |  25  |   1061.32   |   1018.21   |    43.10    |  22335.19   |
 |  26  |   1061.32   |   1016.25   |    45.06    |  22290.13   |
 |  27  |   1061.32   |   1014.20   |    47.12    |  22243.01   |
 |  28  |   1061.32   |   1012.06   |    49.26    |  22193.76   |
 |  29  |   1061.32   |   1009.82   |    51.50    |  22142.25   |
 |  30  |   1061.32   |   1007.47   |    53.84    |  22088.41   |
 |  31  |   1061.32   |   1005.02   |    56.29    |  22032.12   |
 |  32  |   1061.32   |   1002.46   |    58.85    |  21973.26   |
 |  33  |   1061.32   |   999.78    |    61.53    |  21911.73   |
 |  34  |   1061.32   |   996.98    |    64.33    |  21847.40   |
 |  35  |   1061.32   |   994.06    |    67.26    |  21780.14   |
 |  36  |   1061.32   |   991.00    |    70.32    |  21709.82   |
 |  37  |   1061.32   |   987.80    |    73.52    |  21636.30   |
 |  38  |   1061.32   |   984.45    |    76.86    |  21559.43   |
 |  39  |   1061.32   |   980.95    |    80.36    |  21479.07   |
 |  40  |   1061.32   |   977.30    |    84.02    |  21395.05   |
 |  41  |   1061.32   |   973.47    |    87.84    |  21307.21   |
 |  42  |   1061.32   |   969.48    |    91.84    |  21215.37   |
 |  43  |   1061.32   |   965.30    |    96.02    |  21119.36   |
 |  44  |   1061.32   |   960.93    |   100.39    |  21018.97   |
 |  45  |   1061.32   |   956.36    |   104.95    |  20914.02   |
 |  46  |   1061.32   |   951.59    |   109.73    |  20804.29   |
 |  47  |   1061.32   |   946.60    |   114.72    |  20689.57   |
 |  48  |   1061.32   |   941.38    |   119.94    |  20569.63   |
 |  49  |   1061.32   |   935.92    |   125.40    |  20444.23   |
 |  50  |   1061.32   |   930.21    |   131.10    |  20313.13   |
 |  51  |   1061.32   |   924.25    |   137.07    |  20176.06   |
 |  52  |   1061.32   |   918.01    |   143.31    |  20032.75   |
 |  53  |   1061.32   |   911.49    |   149.83    |  19882.93   |
 |  54  |   1061.32   |   904.67    |   156.64    |  19726.28   |
 |  55  |   1061.32   |   897.55    |   163.77    |  19562.51   |
 |  56  |   1061.32   |   890.09    |   171.22    |  19391.29   |
 |  57  |   1061.32   |   882.30    |   179.01    |  19212.28   |
 |  58  |   1061.32   |   874.16    |   187.16    |  19025.12   |
 |  59  |   1061.32   |   865.64    |   195.67    |  18829.45   |
 |  60  |   1061.32   |   856.74    |   204.58    |  18624.87   |
 |  61  |   1061.32   |   847.43    |   213.88    |  18410.99   |
 |  62  |   1061.32   |   837.70    |   223.62    |  18187.37   |
 |  63  |   1061.32   |   827.53    |   233.79    |  17953.58   |
 |  64  |   1061.32   |   816.89    |   244.43    |  17709.15   |
 |  65  |   1061.32   |   805.77    |   255.55    |  17453.60   |
 |  66  |   1061.32   |   794.14    |   267.18    |  17186.42   |
 |  67  |   1061.32   |   781.98    |   279.33    |  16907.09   |
 |  68  |   1061.32   |   769.27    |   292.04    |  16615.05   |
 |  69  |   1061.32   |   755.98    |   305.33    |  16309.71   |
 |  70  |   1061.32   |   742.09    |   319.22    |  15990.49   |
 |  71  |   1061.32   |   727.57    |   333.75    |  15656.74   |
 |  72  |   1061.32   |   712.38    |   348.93    |  15307.81   |
 |  73  |   1061.32   |   696.51    |   364.81    |  14943.00   |
 |  74  |   1061.32   |   679.91    |   381.41    |  14561.59   |
 |  75  |   1061.32   |   662.55    |   398.76    |  14162.82   |
 |  76  |   1061.32   |   644.41    |   416.91    |  13745.91   |
 |  77  |   1061.32   |   625.44    |   435.88    |  13310.04   |
 |  78  |   1061.32   |   605.61    |   455.71    |  12854.33   |
 |  79  |   1061.32   |   584.87    |   476.44    |  12377.88   |
 |  80  |   1061.32   |   563.19    |   498.12    |  11879.76   |
 |  81  |   1061.32   |   540.53    |   520.79    |  11358.97   |
 |  82  |   1061.32   |   516.83    |   544.48    |  10814.49   |
 |  83  |   1061.32   |   492.06    |   569.26    |  10245.23   |
 |  84  |   1061.32   |   466.16    |   595.16    |   9650.08   |
 |  85  |   1061.32   |   439.08    |   622.24    |   9027.84   |
 |  86  |   1061.32   |   410.77    |   650.55    |   8377.29   |
 |  87  |   1061.32   |   381.17    |   680.15    |   7697.14   |
 |  88  |   1061.32   |   350.22    |   711.10    |   6986.04   |
 |  89  |   1061.32   |   317.86    |   743.45    |   6242.59   |
 |  90  |   1061.32   |   284.04    |   777.28    |   5465.31   |
 |  91  |   1061.32   |   248.67    |   812.64    |   4652.67   |
 |  92  |   1061.32   |   211.70    |   849.62    |   3803.05   |
 |  93  |   1061.32   |   173.04    |   888.28    |   2914.77   |
 |  94  |   1061.32   |   132.62    |   928.69    |   1986.08   |
 |  95  |   1061.32   |    90.37    |   970.95    |   1015.13   |
 |  96  |   1061.32   |    46.19    |   1015.13   |    0.00     |
 |______|_____________|_____________|_____________|_____________|
 |Total |  101886.36  |  78886.36   |  23000.00   |      0      |
 |______|_____________|_____________|_____________|_____________|
```

Figura 3: Calculadora CDC - saída.

Desta forma, é mais fácil criar testes específicos para cada método. Abaixo, está impressa a saída da utilização da interface textual, na simulação real de um empréstimo do Banco do Brasil contraído em 2019, com 96 parcelas mensais e taxa de 4.98% ao mês. Um claro exemplo de agiotagem explícita: 383% em oito anos.

```
roma:$ _02f_rational.py --parcelas=96 --taxa=4.55 --valorV=23000
                                                  --valorP=111064.80

O preço à vista é menor ou igual do que preço total corrigido.
Taxa Real = 4.9829%, Iterações = 3, Fator = 0.2257
Preço à vista + juros de 4.55% ao mês = $101886.36
Preço a prazo - juros de 4.55% ao mês = $25071.96
Juros Embutidos = ($111064.80 - $23000.00) / $23000.00 * 100 = 382.89%
Desconto = ($111064.80 - $23000.00) / $111064.80 * 100 = 79.29%
Excesso = $25071.96 - $23000.00 = $2071.96
Excesso = ($111064.80 - $101886.36) * 0.2257 = $2071.96
Percentual pago a mais = 8.26%

Coeficiente de Financiamento: 0.046144
Prestação: $1061.32

Note-se que taxa informada foi menor do que a taxa real.
```


BWK12A9 • Máquina de Lavar

12Kg titânio com Ciclo Tira Manchas Advanced e Ciclo Antibolinha • 110V

Voltagem: [✓ 110V] 220V

~~R$ 2.479,00~~ 15%

R$ 1.889,10 à vista (10% de desconto)

ou R$ 2.099,00 em 12x de R$ 174,91 s/juros

20x de R$ 104,95 s/juros no Cartão Brastemp

Frete Grátis* Exceto norte Cupom CASA100

Aproveite até 21x s/juros e benefícios exclusivos. Saiba mais

Ver todas as formas de pagamento

Digite seu CEP calcule o frete Não sei o CEP

00000-000

Figura 4: Lavadora Brastemp.

Considere-se, agora, o exemplo da Figura 4: preço à vista = R$ 1889.10, e a prazo = R$ 2099.00, quando parcelado em 12 vezes sem juros. Nesse caso, o juros real é de 1.66% ao mês (contra 0.12% da poupança), e não zero, como sugerido. Por outro lado, o desconto à vista é realmente de 10% em um ano (contra 1.40% da poupança).

```
Usage ./_02f_rational.py -n [nº parcelas] -t [taxa] -x [valor a prazo]
                          -y [valor à vista] -e [entrada] -v [modo verboso]

roma:$ _02f_rational.py -n12 -t0 -x2099 -y1889.10 -v

Taxa = 1.6594 % - 7 iterações

O preço à vista é igual ao preço total corrigido.
Taxa Real = 1.6594%, Iterações = 7, Fator = 0.9000
Preço à vista + juros de 1.66% ao mês = $2099.00
Preço a prazo - juros de 1.66% ao mês = $1889.10
Juros Embutidos = ($2099.00 - $1889.10) / $1889.10 * 100 = 11.11%
Desconto = ($2099.00 - $1889.10) / $2099.00 * 100 = 10.00%
Excesso = $1889.10 - $1889.10 = $-0.00
Excesso = ($2099.00 - $2099.00) * 0.9000 = $-0.00
Percentual pago a mais = -0.00%

Coeficiente de Financiamento: 0.092593
Prestação: $174.92
```

```
                                    Tabela Price
             ______________________________________________________________
            | Mês  |  Prestação   |    Juros     | Amortização |Saldo Devedor|
            |______|______________|______________|_____________|_____________|
            |  1   |    174.92    |    31.35     |   143.57    |   1745.53   |
            |  2   |    174.92    |    28.96     |   145.95    |   1599.58   |
            |  3   |    174.92    |    26.54     |   148.37    |   1451.20   |
            |  4   |    174.92    |    24.08     |   150.84    |   1300.37   |
            |  5   |    174.92    |    21.58     |   153.34    |   1147.03   |
            |  6   |    174.92    |    19.03     |   155.88    |   991.15    |
            |  7   |    174.92    |    16.45     |   158.47    |   832.68    |
            |  8   |    174.92    |    13.82     |   161.10    |   671.58    |
            |  9   |    174.92    |    11.14     |   163.77    |   507.80    |
            |  10  |    174.92    |    8.43      |   166.49    |   341.31    |
            |  11  |    174.92    |    5.66      |   169.25    |   172.06    |
            |  12  |    174.92    |    2.86      |   172.06    |    0.00     |
            |______|______________|______________|_____________|_____________|
            |Total |   2099.00    |   209.90     |   1889.10   |      0      |
            |______|______________|______________|_____________|_____________|
```

Código 2: Calculadora - saída modo texto.

Se você utilizou a API sugerida na AD1, o código 1 rodará sem modificação alguma. Caso, contrário, será necessário adaptar as chamadas dos métodos utilizados. De qualquer forma, todo o seu código deverá estar documentado de acordo com o Doxygen[5] e seguir as diretivas do PEP8[6] [7].

[5]https://www.doxygen.nl/index.html

[6]https://realpython.com/python-pep8/

[7]pep8 _02f_rational.py –ignore=E401,E501,E266,E226,E302,E221

3 Tarefas complementares

1. É comum que empresas de desenvolvimento de software obriguem que testes sejam escritos, antes da implementação de cada método. Nesta tarefa, usaremos o unittest[8], para escrever testes unitários, que basicamente comparam os resultados dos cálculos, após a execução de um método.

2. Crie um arquivo calcTest.py (código 3) com uma classe que utiliza o unittest[9] para testar cada método/função da sua calculadora.

```
#!/usr/bin/env python
# coding: UTF-8
#
## @package calcTest
#
#  Class for testing the Calculator.
#
#  @author Paulo Roma
#  @since 23/08/2021
#  @see https://docs.python.org/2/library/unittest.html
#

from _02f_rational import getInterest, CF, priceTable, setDownPayment
import sys
import unittest

##
# Class for testing certain aspects of the behavior of
# the calculator.
#
class CalcTest(unittest.TestCase):
    ##
    # setUp is called automatically before every test is executed.
    #
    def setUp(self):
        setDownPayment(False)

    ## Test getInterest.
    def testTaxa(self):
        """_02f_rational.py -n 96 -t 0 -x 134788.8 -y 63816.24"""

        t, iter = getInterest(134788.8, 63816.24, 96)
        self.assertEqual(round(t, 4), 1.8052)

    ## Test priceTable.
    def testPriceTable(self):
        """_02f_rational.py -n 12 -t 0 -x 134788.8 -y 63816.24 -v"""

        np = 12
```

[8]https://www.youtube.com/watch?v=6tNS--WetLI

[9]https://docs.python.org/3/library/unittest.html

```
        pp = 134788.8
        pv = 63816.24
        t, iter = getInterest(pp, pv, np)
        t *= 0.01
        cf = CF(t, np)
        pmt = pv * cf
        pt = priceTable(np, pv, t, pmt)

        # autopep8: off
        table = [['Mês', 'Prestação', 'Juros', 'Amortização', 'Saldo Devedor'],
        [1,  11232.400178356344, 8880.06618026338,   2352.333998092963,  61463.90600190703],
        [2,  11232.400178356344, 8552.737563266373,  2679.66261508997,   58784.24338681706],
        [3,  11232.400178356344, 8179.860982590724,  3052.5391957656193, 55731.704191051445],
        [4,  11232.400178356344, 7755.098413121783,  3477.3017652345607, 52254.40242581689],
        [5,  11232.400178356344, 7271.229889936626,  3961.1702884197175, 48293.23213739717],
        [6,  11232.400178356344, 6720.030785876906,  4512.369392479438,  43780.86274491774],
        [7,  11232.400178356344, 6092.132012225994,  5140.2681661303495, 38640.594578787386],
        [8,  11232.400178356344, 5376.8607662306395, 5855.539412125704,  32785.05516666168],
        [9,  11232.400178356344, 4562.059118549473,  6670.34105980687,   26114.71410685481],
        [10, 11232.400178356344, 3633.8773570415374, 7598.522821314806,  18516.191285540008],
        [11, 11232.400178356344, 2576.5385742251715, 8655.861604131172,  9860.329681408835],
        [12, 11232.400178356344, 1372.0704969475762, 9860.329681408768,6.730260793119669e-11],
        ['Total', 134788.80214027612, 70972.56214027619, 63816.23999999995, 0]]
        # autopep8: on

        self.assertEqual(pt, table)

    ## Another test.
    # def test...


if __name__ == "__main__":
    unittest.main()
```

Código 3: calcTest.py