Programs related to prime arclength prediction trials

```
x1 =1381804.899 (* insira o valor de x1 aqui *);
x = 2965178.021(* insira o valor de x aqui *);
y1 =263881(* insira o valor de y1 aqui *);
T = 10.4729(* insira o valor de T aqui *);
z =x*T (* insira o valor de z aqui *);(* Definindo a equação *)
g=z/x1
g1=N[g,9]
dd=g1/2
jk=y1*dd
g1/T
equacao = (x1 * x) / (y1 * y) == T * t1;
(* Usando Solve para encontrar y e t1 *)
solucao =y/.FindInstance[{equacao, t1 == g1 / 2}, {y, t1}]
solucao2 =t1/.FindInstance[{equacao, t1 == g1 / 2}, {y, t1}]
(* Exibindo a solução *)
a=solucao
b=solucao2
y1*b
(* Definindo os valores *)
x1 = 1381804.899;
x = 2965178.021;
y1 = 263881;
T = 10.4729;
z = x * T;
g = z / x1;
g1 = N[g, 9];
dd = g1 / 2;

(* Criando a tabela *)
tabela = Table[
    {
      "Denominador" -> i,
      "dd" -> g1 / i,
      "jk" -> y1 * (g1 / i)
     },
    {i, 1, 10} (* Variando o denominador de 1 a 10 *)
  ];

(* Exibindo a tabela *)
tabela
```

22.4735

22.4735

11.2368

2.96517*10^6

2.14587

{131942.}

{11.2368}

{131942.}

{11.2368}

{2.96517*10^6}

{{Denominador->1,dd->22.4735,jk->5.93033*10^6},{Denominador->2,dd->11.2368,jk->2.96517*10^6},{Denominador->3,dd->7.49117,jk->1.97678*10^6},{Denominador->4,dd->5.61838,jk->1.48258*10^6},{Denominador->5,dd->4.4947,jk->1.18607*10^6},{Denominador->6,dd->3.74559,jk->988389.},{Denominador->7,dd->3.2105,jk->847191.},{Denominador->8,dd->2.80919,jk->741292.},{Denominador->9,dd->2.49706,jk->658926.},{Denominador->10,dd->2.24735,jk->593033.}}

```
(* Definindo a função *)

f[x_] := 6.597032


(* Calculando a integral definida *)

valorIntegral = Integrate[f[x], {x, -a, a}]


(* Igualando ao valor dado e resolvendo para a *)

solucao= Solve[valorIntegral == 1976777.8560, a]


(* Definindo a função constante *)

f[x_] := 6.597032


(* Comprimento de arco para uma função constante entre -a e a *)

comprimentoArco = 2 *b * f[x]


(* Igualando ao valor dado e resolvendo para a *)

solucao1=Solve[comprimentoArco == 5930333,b]

a1=a /. solucao[[1]]

b1=b /. solucao1[[1]]
```

c=b1/a1

13.1941 a

{{a->149823.}}

13.1941 b

{{b->449470.}}

149823.

449470.

3.

c = 6.597032310;

(* Given value of n *)

n = 15000;

(* Calculate the definite integral from -n to n *)

desiredIntegral = c * 2 * n;

(* Print the result *)

Print["The definite integral from -", n, " to ", n, " is ", desiredIntegral];

The definite integral from -15000 to 15000 is 197911.

(* Given constant c *)

c = 6.597032310;

(* Desired definite integral value *)

desiredIntegral = 149823;

(* Function to calculate the definite integral from -n to n *)

CalculateIntegral[n_] := c * 2 * n

```
(* Solve for n *)
n = desiredIntegral / (c * 2);

(* Print the interval *)
Print["The interval [-", n, ", ", n, "] results in a definite integral of ", desiredIntegral];
```

The interval [-11355.3, 11355.3] results in a definite integral of 149823

```
c = 6.597032310;
(* Lista de números primos nas posições desejadas *)
primos = {120689, 163841, 224737, 15485863, 999999000001};

(* Função para calcular a integral definida de -n até n *)
CalcularIntegral[n_] := c * 2 * n

(* Calcular e imprimir as integrais para cada número primo *)
resultados = Table[
  n = primo;
  integral = CalcularIntegral[n];
  Print["A integral definida de -", n, " até ", n, " é ", integral];
  integral,
  {primo, primos}
]

(* Ajuste para calcular a integral1 *)
primos = {120689, 163841, 224737, 15485863, 999999000001};
primos = PrimePi[primos];

resultados1 = Table[
  n = primo;
  integral1 = CalcularIntegral[n];
  Print["A integral definida de -", n, " até ", n, " é ", integral1];
```

```
  integral1,
  {primo, primos}
]

(* Calcular e imprimir as divisões *)
divisoes = resultados / resultados1;
Print["Os valores das divisões são: ", divisoes];
a=divisoes
resultados1/(2*c)
```

(Debug) During evaluation of In[51]:= A integral definida de -120689 até 120689 é 1.59238*10^6

(Debug) During evaluation of In[51]:= A integral definida de -163841 até 163841 é 2.16173*10^6

(Debug) During evaluation of In[51]:= A integral definida de -224737 até 224737 é 2.96519*10^6

(Debug) During evaluation of In[51]:= A integral definida de -15485863 até 15485863 é 2.04321*10^8

(Debug) During evaluation of In[51]:= A integral definida de -999999000001 até 999999000001 é 1.31941*10^13

(Debug) Out[54]= {1.59238*10^6,2.16173*10^6,2.96519*10^6,2.04321*10^8,1.31941*10^13}

(Debug) During evaluation of In[51]:= A integral definida de -11357 até 11357 é 149845.

(Debug) During evaluation of In[51]:= A integral definida de -15000 até 15000 é 197911.

(Debug) During evaluation of In[51]:= A integral definida de -20000 até 20000 é 263881.

(Debug) During evaluation of In[51]:= A integral definida de -1000000 até 1000000 é 1.31941*10^7

(Debug) During evaluation of In[51]:= A integral definida de -37607875619 até 37607875619 é 4.96201*10^11

(Debug) Out[57]= {149845.,197911.,263881.,1.31941*10^7,4.96201*10^11}

(Debug) During evaluation of In[51]:= Os valores das divisões são: {10.6268,10.9227,11.2368,15.4859,26.5901}

(Debug) Out[60]= {10.6268,10.9227,11.2368,15.4859,26.5901}

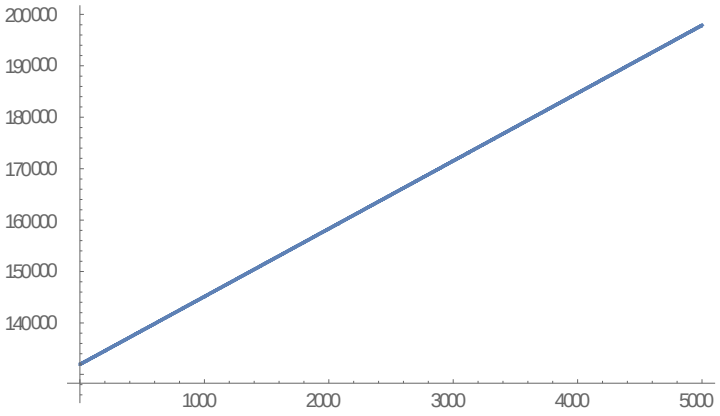
(Debug) Out[61]= {11357.,15000.,20000.,1.*10^6,3.76079*10^10}

A Linear Graph from

```
(* Given constant c *)

c = 6.597032310;


(* Given value of n *)

n = Range[10000,15000];


(* Calculate the definite integral from -n to n *)

desiredIntegral = c * 2 * n
```

{131941.,131954.,131967.,131980.,131993.,132007.,132020.,132033.,132046.,132059.,13207
3.,132086.,132099.,132112.,132125.,132139.,132152.,132165.,132178.,132191.,132205.,1322
18.,132231.,132244.,132257.,132270.,132284.,132297.,132310.,132323.,132336.,132350.,132
363.,132376.,132389.,132402.,132416.,132429.,132442.,132455.,132468.,132482.,132495.,13
2508.,132521.,132534.,132548.,132561.,132574.,132587.,132600.,132614.,132627.,132640.,1
32653.,132666.,132680.,132693.,132706.,132719.,132732.,132745.,132759.,132772.,132785.,
132798.,132811.,132825.,132838.,132851.,132864.,132877.,132891.,132904.,132917.,132930
.,132943.,132957.,132970.,132983.,132996.,133009.,133023.,133036.,133049.,133062.,13307
5.,133089.,133102.,133115.,133128.,133141.,133155.,133168.,133181.,133194.,133207.,1332
20.,133234.,133247.,133260.,133273.,133286.,133300.,133313.,133326.,133339.,133352.,133
366.,133379.,133392.,133405.,133418.,133432.,133445.,133458.,133471.,133484.,133498.,13
3511.,133524.,133537.,133550.,133564.,133577.,133590.,133603.,133616.,133629.,133643.,1
33656.,133669.,133682.,133695.,133709.,133722.,133735.,133748.,133761.,133775.,133788.,
133801.,133814.,133827.,133841.,133854.,133867.,133880.,133893.,133907.,133920.,133933
.,133946.,133959.,133973.,133986.,133999.,134012.,134025.,134039.,134052.,134065.,13407
8.,134091.,134104.,134118.,134131.,134144.,134157.,134170.,134184.,134197.,134210.,1342
23.,134236.,134250.,134263.,134276.,134289.,134302.,134316.,134329.,134342.,134355.,134
368.,134382.,134395.,134408.,134421.,134434.,134448.,134461.,134474.,134487.,134500.,13
4513.,134527.,134540.,134553.,134566.,134579.,134593.,134606.,134619.,134632.,134645.,1
34659.,134672.,134685.,134698.,134711.,134725.,134738.,134751.

```
(* Print the result *)

Print["The definite integral from -", n, " to ", n, " is ", desiredIntegral];


ListLinePlot[desiredIntegral]
```



mathematica

(Definindo a sequência dada)

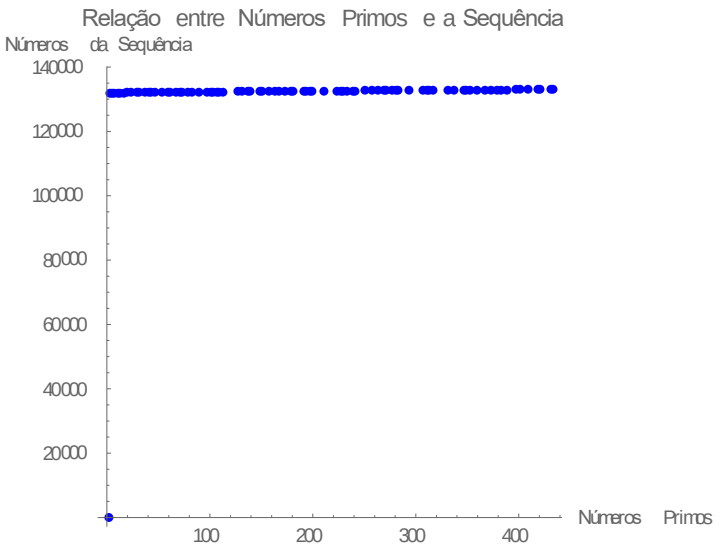
```
sequencia = {0, 131941, 131954, 131967, 131980, 131993, 132007, 132020, 132033, 132046,
        132059, 132073, 132086, 132099, 132112, 132125, 132139, 132152, 132165,
        132178, 132191, 132205, 132218, 132231, 132244, 132257, 132270, 132284,
        132297, 132310, 132323, 132336, 132350, 132363, 132376, 132389, 132402,
        132416, 132429, 132442, 132455, 132468, 132482, 132495, 132508, 132521,
        132534, 132548, 132561, 132574, 132587, 132600, 132614, 132627, 132640,
        132653, 132666, 132680, 132693, 132706, 132719, 132732, 132745, 132759,
        132772, 132785, 132798, 132811, 132825, 132838, 132851, 132864, 132877,
        132891, 132904, 132917, 132930, 132943, 132957, 132970, 132983, 132996,
        133009, 133023};
```

(Encontrando os números primos até o último número da sequência)

```
primos = Select[Range[Max[sequencia]], PrimeQ];
```

(Criando um gráfico da relação entre os números primos e a sequência)

```
ListPlot[Transpose[{primos[[1 ;; Length[sequencia]]], sequencia}],
 PlotStyle -> {PointSize[Medium], Blue},
 AxesLabel -> {"Números Primos", "Números da Sequência"},
 PlotLabel -> "Relação entre Números Primos e a Sequência",
 AspectRatio -> 1]
```



(* Dado constante c *)

c = 6.597032310;

(* Dado valor de n como os números primos de 10000 a 15000 *)

n = Prime[Range[10000, 15000]];

(* Calcular a integral definida de -n a n *)

desiredIntegral = c * 2 * n;

(* Imprimir o resultado *)

Print["A integral definida de -", n, " a ", n, " é ", desiredIntegral];

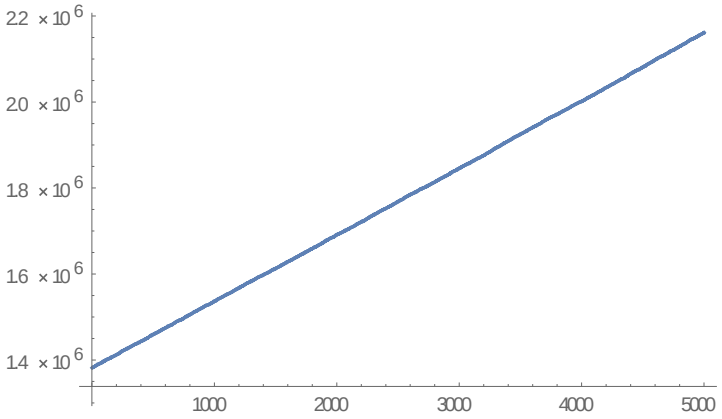
(* Plotar o resultado *)

ListLinePlot[desiredIntegral]

{104729,104743,104759,104761,104773,104779,104789,104801,104803,104827,104831,1048
49,104851,104869,104879,104891,104911,104917,104933,104947,104953,104959,104971,10
4987,104999,105019,105023,105031,105037,105071,105097,105107,105137,105143,105167,
105173,105199,105211,105227,105229,105239,105251,105253,105263,105269,105277,1053
19,105323,105331,105337,105341,105359,105361,105367,105373,105379,105389,105397,10
5401,105407,105437,105449,105467,105491,105499,105503,105509,105517,105527,105529,
105533,105541,105557,105563,105601,105607,105613,105619,105649,105653,105667,1056
73,105683,105691,105701,105727,105733,105751,105761,105767,105769,105817,105829,10
5863,105871,105883,105899,105907,105913,105929,105943,105953,105967,105971,105977,
105983,105997,106013,106019,106031,106033,106087,106103,106109,106121,106123,1061

29,106163,106181,106187,106189,106207,106213,106217,106219,106243,106261,106273,106277,106279,106291,106297,106303,106307,106319,106321,106331,106349,106357,106363,106367,106373,106391,106397,106411,106417,106427,106433,106441,106451,106453,106487,106501,106531,106537,106541,106543,106591,106619,106621,106627,106637,106649,106657,106661,106663,106669,106681,106693,106699,106703,106721,106727,106739,106747,106751,106753,106759,106781,106783,106787,106801,106823,106853,106859,106861,106867,106871,106877,106903,106907,106921,106937,106949,106957,106961,106963,106979, É

{1.3818*10^6,1.38199*10^6,1.3822*10^6,1.38222*10^6,1.38238*10^6,1.38246*10^6,1.38259*10^6,1.38275*10^6,1.38278*10^6,1.38309*10^6,1.38315*10^6,1.38338*10^6,1.38341*10^6,1.38365*10^6,1.38378*10^6,1.38394*10^6,1.3842*10^6,1.38428*10^6,1.38449*10^6,1.38468*10^6,1.38476*10^6,1.38484*10^6,1.38499*10^6,1.38521*10^6,1.38536*10^6,1.38563*10^6,1.38568*10^6,1.38579*10^6,1.38586*10^6,1.38631*10^6,1.38666*10^6,1.38679*10^6,1.38718*10^6,1.38726*10^6,1.38758*10^6,1.38766*10^6,1.388*10^6,1.38816*10^6,1.38837*10^6,1.3884*10^6,1.38853*10^6,1.38869*10^6,1.38871*10^6,1.38885*10^6,1.38893*10^6,1.38903*10^6,1.38959*10^6,1.38964*10^6,1.38974*10^6,1.38982*10^6,1.38988*10^6,1.39011*10^6,1.39014*10^6,1.39022*10^6,1.3903*10^6,1.39038*10^6,1.39051*10^6,1.39061*10^6,1.39067*10^6,1.39075*10^6,1.39114*10^6,1.3913*10^6,1.39154*10^6,1.39186*10^6,1.39196*10^6,1.39201*10^6,



c = 6.597032310;

(* Dado valor de n como os números primos de 10000 a 15000 *)

n = Prime[Range[10000, 15000]];

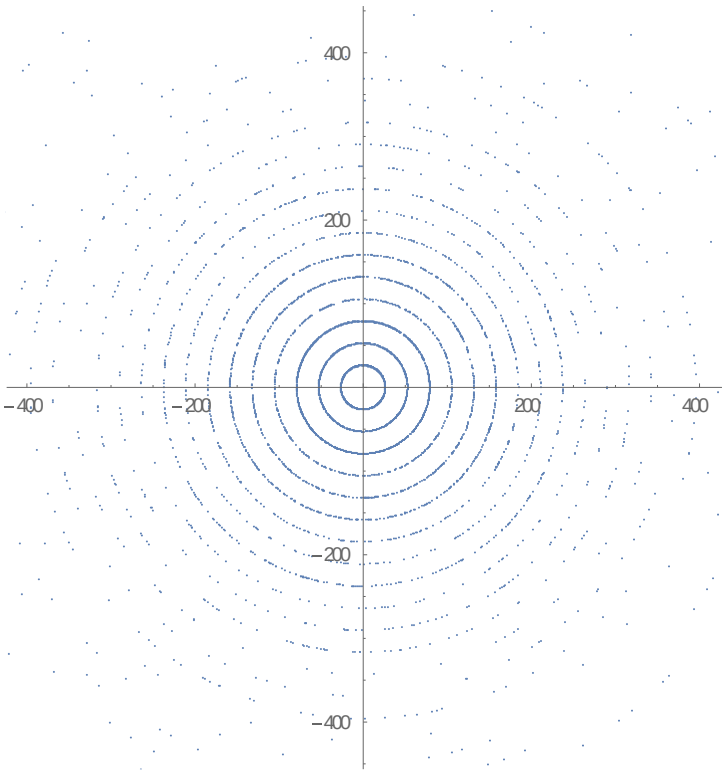
(* Calcular a integral definida de -n a n *)

desiredIntegral = c * 2 * n;

as=Differences[%]

(* Imprimir o resultado *)

```
Print["A integral definida de -", n, " a ", n, " é ", desiredIntegral];


(* Plotar o resultado *)

ListLinePlot[desiredIntegral]

ListPolarPlot[as]
```

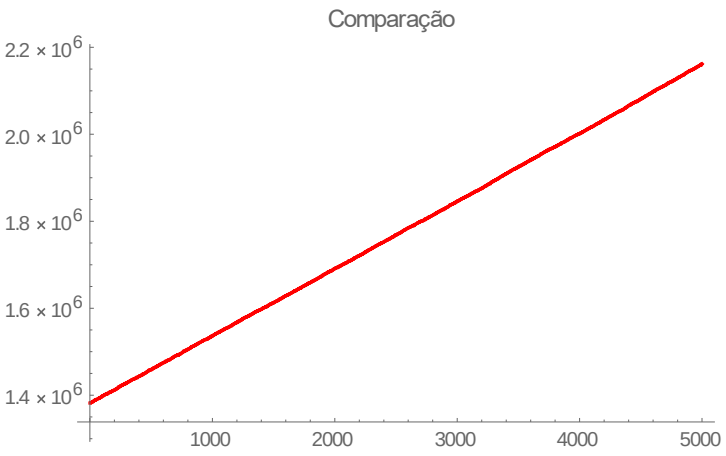



```
(* Programa 1 *)

c = 6.597032310;

n1 = Prime[Range[10000, 15000]];

desiredIntegral1 = c * 2 * n1;


(* Programa 2 *)

n2 = Range[10000, 15000];

desiredIntegral2 = c * 2 * n2;


(* Mapeamento de Índices *)
```

```
index = 500; (* Exemplo de índice *)

valorPrimo = desiredIntegral1[[index]];

valorInteiro = desiredIntegral2[[index]];


Print["Para o índice ", index, ", o valor primo é ", valorPrimo, " e o valor inteiro é ",
valorInteiro];


(* Comparação *)

Show[

  ListLinePlot[desiredIntegral1, PlotStyle -> Red, PlotLabel -> "Comparação"],

  ListLinePlot[desiredIntegral2, PlotStyle -> Blue]

]
```

(Debug) During evaluation of In[1]:= Para o índice 500, o valor primo é 1.45909*10^6 e o valor inteiro é 138524.

(Debug) Out[9]=




```
(* Definindo a função dn (substitua pela definição correta) *)

dn[u_, m_] := 2*6.597032310*y


(* Função para calcular a integral numericamente *)

integral[x_, m_] := NIntegrate[dn[y/x, m], {y, -x, x}]


(* Função para encontrar soluções (abordagem simplificada) *)
```

```
findSolutions[maxPrime_] :=
 Module[{primes = Prime[Range[maxPrime]], solutions = {}},
  Do[
   If[integral[x, m] == integral[Prime[x], m],
    AppendTo[solutions, {x, Prime[x]}]
    ],
   {x, maxPrime}
   ];
  solutions
  ]

(* Exemplo de uso *)
m = 0.5; (* Parâmetro da função dn *)
maxPrime = 1000; (* Procurar soluções até o 100º primo *)
solutions = findSolutions[maxPrime]
```

```
{{1, 2}, {2, 3}, {3, 5}, {4, 7}, {5, 11}, {6, 13}, {7, 17}, {8,
  19}, {9, 23}, {10, 29}, {11, 31}, {12, 37}, {13, 41}, {14, 43}, {15,
  47}, {16, 53}, {17, 59}, {18, 61}, {19, 67}, {20, 71}, {21,
  73}, {22, 79}, {23, 83}, {24, 89}, {25, 97}, {26, 101}, {27,
  103}, {28, 107}, {29, 109}, {30, 113}, {31, 127}, {32, 131}, {33,
  137}, {34, 139}, {35, 149}, {36, 151}, {37, 157}, {38, 163}, {39,
  167}, {40, 173}, {41, 179}, {42, 181}, {43, 191}, {44, 193}, {45,
  197}, {46, 199}, {47, 211}, {48, 223}, {49, 227}, {50, 229}, {51,
  233}, {52, 239}, {53, 241}, {54, 251}, {55, 257}, {56, 263}, {57,
  269}, {58, 271}, {59, 277}, {60, 281}, {61, 283}, {62, 293}, {63,
  307}, {64, 311}, {65, 313}, {66, 317}, {67, 331}, {68, 337}, {69,
  347}, {70, 349}, {71, 353}, {72, 359}, {73, 367}, {74, 373}, {75,
  379}, {76, 383}, {77, 389}, {78, 397}, {79, 401}, {80, 409}, {81,
  419}, {82, 421}, {83, 431}, {84, 433}, {85, 439}, {86, 443}, {87,
  449}, {88, 457}, {89, 461}, {90, 463}, {91, 467}, {92, 479}, {93,
```

487}, {94, 491}, {95, 499}, {96, 503}, {97, 509}, {98, 521}, {99,
523}, {100, 541}, {101, 547}, {102, 557}, {103, 563}, {104,
569}, {105, 571}, {106, 577}, {107, 587}, {108, 593}, {109,
599}, {110, 601}, {111, 607}, {112, 613}, {113, 617}, {114,
619}, {115, 631}, {116, 641}, {117, 643}, {118, 647}, {119,
653}, {120, 659}, {121, 661}, {122, 673}, {123, 677}, {124,
683}, {125, 691}, {126, 701}, {127, 709}, {128, 719}, {129,
727}, {130, 733}, {131, 739}, {132, 743}, {133, 751}, {134,
757}, {135, 761}, {136, 769}, {137, 773}, {138, 787}, {139,
797}, {140, 809}, {141, 811}, {142, 821}, {143, 823}, {144,
827}, {145, 829}, {146, 839}, {147, 853}, {148, 857}, {149,
859}, {150, 863}, {151, 877}, {152, 881}, {153, 883}, {154,
887}, {155, 907}, {156, 911}, {157, 919}, {158, 929}, {159,
937}, {160, 941}, {161, 947}, {162, 953}, {163, 967}, {164,
971}, {165, 977}, {166, 983}, {167, 991}, {168, 997}, {169,
1009}, {170, 1013}, {171, 1019}, {172, 1021}, {173, 1031}, {174,
1033}, {175, 1039}, {176, 1049}, {177, 1051}, {178, 1061}, {179,
1063}, {180, 1069}, {181, 1087}, {182, 1091}, {183, 1093}, {184,
1097}, {185, 1103}, {186, 1109}, {187, 1117}, {188, 1123}, {189,
1129}, {190, 1151}, {191, 1153}, {192, 1163}, {193, 1171}, {194,
1181}, {195, 1187}, {196, 1193}, {197, 1201}, {198, 1213}, {199,
1217}, {200, 1223}, {201, 1229}, {202, 1231}, {203, 1237}, {204,
1249}, {205, 1259}, {206, 1277}, {207, 1279}, {208, 1283}, {209,
1289}, {210, 1291}, {211, 1297}, {212, 1301}, {213, 1303}, {214,
1307}, {215, 1319}, {216, 1321}, {217, 1327}, {218, 1361}, {219,
1367}, {220, 1373}, {221, 1381}, {222, 1399}, {223, 1409}, {224,
1423}, {225, 1427}, {226, 1429}, {227, 1433}, {228, 1439}, {229,
1447}, {230, 1451}, {231, 1453}, {232, 1459}, {233, 1471}, {234,
1481}, {235, 1483}, {236, 1487}, {237, 1489}, {238, 1493}, {239,
1499}, {240, 1511}, {241, 1523}, {242, 1531}, {243, 1543}, {244,
1549}, {245, 1553}, {246, 1559}, {247, 1567}, {248, 1571}, {249,

1579}, {250, 1583}, {251, 1597}, {252, 1601}, {253, 1607}, {254,
1609}, {255, 1613}, {256, 1619}, {257, 1621}, {258, 1627}, {259,
1637}, {260, 1657}, {261, 1663}, {262, 1667}, {263, 1669}, {264,
1693}, {265, 1697}, {266, 1699}, {267, 1709}, {268, 1721}, {269,
1723}, {270, 1733}, {271, 1741}, {272, 1747}, {273, 1753}, {274,
1759}, {275, 1777}, {276, 1783}, {277, 1787}, {278, 1789}, {279,
1801}, {280, 1811}, {281, 1823}, {282, 1831}, {283, 1847}, {284,
1861}, {285, 1867}, {286, 1871}, {287, 1873}, {288, 1877}, {289,
1879}, {290, 1889}, {291, 1901}, {292, 1907}, {293, 1913}, {294,
1931}, {295, 1933}, {296, 1949}, {297, 1951}, {298, 1973}, {299,
1979}, {300, 1987}, {301, 1993}, {302, 1997}, {303, 1999}, {304,
2003}, {305, 2011}, {306, 2017}, {307, 2027}, {308, 2029}, {309,
2039}, {310, 2053}, {311, 2063}, {312, 2069}, {313, 2081}, {314,
2083}, {315, 2087}, {316, 2089}, {317, 2099}, {318, 2111}, {319,
2113}, {320, 2129}, {321, 2131}, {322, 2137}, {323, 2141}, {324,
2143}, {325, 2153}, {326, 2161}, {327, 2179}, {328, 2203}, {329,
2207}, {330, 2213}, {331, 2221}, {332, 2237}, {333, 2239}, {334,
2243}, {335, 2251}, {336, 2267}, {337, 2269}, {338, 2273}, {339,
2281}, {340, 2287}, {341, 2293}, {342, 2297}, {343, 2309}, {344,
2311}, {345, 2333}, {346, 2339}, {347, 2341}, {348, 2347}, {349,
2351}, {350, 2357}, {351, 2371}, {352, 2377}, {353, 2381}, {354,
2383}, {355, 2389}, {356, 2393}, {357, 2399}, {358, 2411}, {359,
2417}, {360, 2423}, {361, 2437}, {362, 2441}, {363, 2447}, {364,
2459}, {365, 2467}, {366, 2473}, {367, 2477}, {368, 2503}, {369,
2521}, {370, 2531}, {371, 2539}, {372, 2543}, {373, 2549}, {374,
2551}, {375, 2557}, {376, 2579}, {377, 2591}, {378, 2593}, {379,
2609}, {380, 2617}, {381, 2621}, {382, 2633}, {383, 2647}, {384,
2657}, {385, 2659}, {386, 2663}, {387, 2671}, {388, 2677}, {389,
2683}, {390, 2687}, {391, 2689}, {392, 2693}, {393, 2699}, {394,
2707}, {395, 2711}, {396, 2713}, {397, 2719}, {398, 2729}, {399,
2731}, {400, 2741}, {401, 2749}, {402, 2753}, {403, 2767}, {404,

2777}, {405, 2789}, {406, 2791}, {407, 2797}, {408, 2801}, {409,
2803}, {410, 2819}, {411, 2833}, {412, 2837}, {413, 2843}, {414,
2851}, {415, 2857}, {416, 2861}, {417, 2879}, {418, 2887}, {419,
2897}, {420, 2903}, {421, 2909}, {422, 2917}, {423, 2927}, {424,
2939}, {425, 2953}, {426, 2957}, {427, 2963}, {428, 2969}, {429,
2971}, {430, 2999}, {431, 3001}, {432, 3011}, {433, 3019}, {434,
3023}, {435, 3037}, {436, 3041}, {437, 3049}, {438, 3061}, {439,
3067}, {440, 3079}, {441, 3083}, {442, 3089}, {443, 3109}, {444,
3119}, {445, 3121}, {446, 3137}, {447, 3163}, {448, 3167}, {449,
3169}, {450, 3181}, {451, 3187}, {452, 3191}, {453, 3203}, {454,
3209}, {455, 3217}, {456, 3221}, {457, 3229}, {458, 3251}, {459,
3253}, {460, 3257}, {461, 3259}, {462, 3271}, {463, 3299}, {464,
3301}, {465, 3307}, {466, 3313}, {467, 3319}, {468, 3323}, {469,
3329}, {470, 3331}, {471, 3343}, {472, 3347}, {473, 3359}, {474,
3361}, {475, 3371}, {476, 3373}, {477, 3389}, {478, 3391}, {479,
3407}, {480, 3413}, {481, 3433}, {482, 3449}, {483, 3457}, {484,
3461}, {485, 3463}, {486, 3467}, {487, 3469}, {488, 3491}, {489,
3499}, {490, 3511}, {491, 3517}, {492, 3527}, {493, 3529}, {494,
3533}, {495, 3539}, {496, 3541}, {497, 3547}, {498, 3557}, {499,
3559}, {500, 3571}, {501, 3581}, {502, 3583}, {503, 3593}, {504,
3607}, {505, 3613}, {506, 3617}, {507, 3623}, {508, 3631}, {509,
3637}, {510, 3643}, {511, 3659}, {512, 3671}, {513, 3673}, {514,
3677}, {515, 3691}, {516, 3697}, {517, 3701}, {518, 3709}, {519,
3719}, {520, 3727}, {521, 3733}, {522, 3739}, {523, 3761}, {524,
3767}, {525, 3769}, {526, 3779}, {527, 3793}, {528, 3797}, {529,
3803}, {530, 3821}, {531, 3823}, {532, 3833}, {533, 3847}, {534,
3851}, {535, 3853}, {536, 3863}, {537, 3877}, {538, 3881}, {539,
3889}, {540, 3907}, {541, 3911}, {542, 3917}, {543, 3919}, {544,
3923}, {545, 3929}, {546, 3931}, {547, 3943}, {548, 3947}, {549,
3967}, {550, 3989}, {551, 4001}, {552, 4003}, {553, 4007}, {554,
4013}, {555, 4019}, {556, 4021}, {557, 4027}, {558, 4049}, {559,

4051}, {560, 4057}, {561, 4073}, {562, 4079}, {563, 4091}, {564,
4093}, {565, 4099}, {566, 4111}, {567, 4127}, {568, 4129}, {569,
4133}, {570, 4139}, {571, 4153}, {572, 4157}, {573, 4159}, {574,
4177}, {575, 4201}, {576, 4211}, {577, 4217}, {578, 4219}, {579,
4229}, {580, 4231}, {581, 4241}, {582, 4243}, {583, 4253}, {584,
4259}, {585, 4261}, {586, 4271}, {587, 4273}, {588, 4283}, {589,
4289}, {590, 4297}, {591, 4327}, {592, 4337}, {593, 4339}, {594,
4349}, {595, 4357}, {596, 4363}, {597, 4373}, {598, 4391}, {599,
4397}, {600, 4409}, {601, 4421}, {602, 4423}, {603, 4441}, {604,
4447}, {605, 4451}, {606, 4457}, {607, 4463}, {608, 4481}, {609,
4483}, {610, 4493}, {611, 4507}, {612, 4513}, {613, 4517}, {614,
4519}, {615, 4523}, {616, 4547}, {617, 4549}, {618, 4561}, {619,
4567}, {620, 4583}, {621, 4591}, {622, 4597}, {623, 4603}, {624,
4621}, {625, 4637}, {626, 4639}, {627, 4643}, {628, 4649}, {629,
4651}, {630, 4657}, {631, 4663}, {632, 4673}, {633, 4679}, {634,
4691}, {635, 4703}, {636, 4721}, {637, 4723}, {638, 4729}, {639,
4733}, {640, 4751}, {641, 4759}, {642, 4783}, {643, 4787}, {644,
4789}, {645, 4793}, {646, 4799}, {647, 4801}, {648, 4813}, {649,
4817}, {650, 4831}, {651, 4861}, {652, 4871}, {653, 4877}, {654,
4889}, {655, 4903}, {656, 4909}, {657, 4919}, {658, 4931}, {659,
4933}, {660, 4937}, {661, 4943}, {662, 4951}, {663, 4957}, {664,
4967}, {665, 4969}, {666, 4973}, {667, 4987}, {668, 4993}, {669,
4999}, {670, 5003}, {671, 5009}, {672, 5011}, {673, 5021}, {674,
5023}, {675, 5039}, {676, 5051}, {677, 5059}, {678, 5077}, {679,
5081}, {680, 5087}, {681, 5099}, {682, 5101}, {683, 5107}, {684,
5113}, {685, 5119}, {686, 5147}, {687, 5153}, {688, 5167}, {689,
5171}, {690, 5179}, {691, 5189}, {692, 5197}, {693, 5209}, {694,
5227}, {695, 5231}, {696, 5233}, {697, 5237}, {698, 5261}, {699,
5273}, {700, 5279}, {701, 5281}, {702, 5297}, {703, 5303}, {704,
5309}, {705, 5323}, {706, 5333}, {707, 5347}, {708, 5351}, {709,
5381}, {710, 5387}, {711, 5393}, {712, 5399}, {713, 5407}, {714,

5413}, {715, 5417}, {716, 5419}, {717, 5431}, {718, 5437}, {719,
5441}, {720, 5443}, {721, 5449}, {722, 5471}, {723, 5477}, {724,
5479}, {725, 5483}, {726, 5501}, {727, 5503}, {728, 5507}, {729,
5519}, {730, 5521}, {731, 5527}, {732, 5531}, {733, 5557}, {734,
5563}, {735, 5569}, {736, 5573}, {737, 5581}, {738, 5591}, {739,
5623}, {740, 5639}, {741, 5641}, {742, 5647}, {743, 5651}, {744,
5653}, {745, 5657}, {746, 5659}, {747, 5669}, {748, 5683}, {749,
5689}, {750, 5693}, {751, 5701}, {752, 5711}, {753, 5717}, {754,
5737}, {755, 5741}, {756, 5743}, {757, 5749}, {758, 5779}, {759,
5783}, {760, 5791}, {761, 5801}, {762, 5807}, {763, 5813}, {764,
5821}, {765, 5827}, {766, 5839}, {767, 5843}, {768, 5849}, {769,
5851}, {770, 5857}, {771, 5861}, {772, 5867}, {773, 5869}, {774,
5879}, {775, 5881}, {776, 5897}, {777, 5903}, {778, 5923}, {779,
5927}, {780, 5939}, {781, 5953}, {782, 5981}, {783, 5987}, {784,
6007}, {785, 6011}, {786, 6029}, {787, 6037}, {788, 6043}, {789,
6047}, {790, 6053}, {791, 6067}, {792, 6073}, {793, 6079}, {794,
6089}, {795, 6091}, {796, 6101}, {797, 6113}, {798, 6121}, {799,
6131}, {800, 6133}, {801, 6143}, {802, 6151}, {803, 6163}, {804,
6173}, {805, 6197}, {806, 6199}, {807, 6203}, {808, 6211}, {809,
6217}, {810, 6221}, {811, 6229}, {812, 6247}, {813, 6257}, {814,
6263}, {815, 6269}, {816, 6271}, {817, 6277}, {818, 6287}, {819,
6299}, {820, 6301}, {821, 6311}, {822, 6317}, {823, 6323}, {824,
6329}, {825, 6337}, {826, 6343}, {827, 6353}, {828, 6359}, {829,
6361}, {830, 6367}, {831, 6373}, {832, 6379}, {833, 6389}, {834,
6397}, {835, 6421}, {836, 6427}, {837, 6449}, {838, 6451}, {839,
6469}, {840, 6473}, {841, 6481}, {842, 6491}, {843, 6521}, {844,
6529}, {845, 6547}, {846, 6551}, {847, 6553}, {848, 6563}, {849,
6569}, {850, 6571}, {851, 6577}, {852, 6581}, {853, 6599}, {854,
6607}, {855, 6619}, {856, 6637}, {857, 6653}, {858, 6659}, {859,
6661}, {860, 6673}, {861, 6679}, {862, 6689}, {863, 6691}, {864,
6701}, {865, 6703}, {866, 6709}, {867, 6719}, {868, 6733}, {869,

6737}, {870, 6761}, {871, 6763}, {872, 6779}, {873, 6781}, {874, 6791}, {875, 6793}, {876, 6803}, {877, 6823}, {878, 6827}, {879, 6829}, {880, 6833}, {881, 6841}, {882, 6857}, {883, 6863}, {884, 6869}, {885, 6871}, {886, 6883}, {887, 6899}, {888, 6907}, {889, 6911}, {890, 6917}, {891, 6947}, {892, 6949}, {893, 6959}, {894, 6961}, {895, 6967}, {896, 6971}, {897, 6977}, {898, 6983}, {899, 6991}, {900, 6997}, {901, 7001}, {902, 7013}, {903, 7019}, {904, 7027}, {905, 7039}, {906, 7043}, {907, 7057}, {908, 7069}, {909, 7079}, {910, 7103}, {911, 7109}, {912, 7121}, {913, 7127}, {914, 7129}, {915, 7151}, {916, 7159}, {917, 7177}, {918, 7187}, {919, 7193}, {920, 7207}, {921, 7211}, {922, 7213}, {923, 7219}, {924, 7229}, {925, 7237}, {926, 7243}, {927, 7247}, {928, 7253}, {929, 7283}, {930, 7297}, {931, 7307}, {932, 7309}, {933, 7321}, {934, 7331}, {935, 7333}, {936, 7349}, {937, 7351}, {938, 7369}, {939, 7393}, {940, 7411}, {941, 7417}, {942, 7433}, {943, 7451}, {944, 7457}, {945, 7459}, {946, 7477}, {947, 7481}, {948, 7487}, {949, 7489}, {950, 7499}, {951, 7507}, {952, 7517}, {953, 7523}, {954, 7529}, {955, 7537}, {956, 7541}, {957, 7547}, {958, 7549}, {959, 7559}, {960, 7561}, {961, 7573}, {962, 7577}, {963, 7583}, {964, 7589}, {965, 7591}, {966, 7603}, {967, 7607}, {968, 7621}, {969, 7639}, {970, 7643}, {971, 7649}, {972, 7669}, {973, 7673}, {974, 7681}, {975, 7687}, {976, 7691}, {977, 7699}, {978, 7703}, {979, 7717}, {980, 7723}, {981, 7727}, {982, 7741}, {983, 7753}, {984, 7757}, {985, 7759}, {986, 7789}, {987, 7793}, {988, 7817}, {989, 7823}, {990, 7829}, {991, 7841}, {992, 7853}, {993, 7867}, {994, 7873}, {995, 7877}, {996, 7879}, {997, 7883}, {998, 7901}, {999, 7907}, {1000, 7919}}

Claro! Vamos analisar o código passo a passo:

1. **Definition of the function** dn:
2. dn[u_, m_] := 2*6.597032310*y

This function dn is defined to receive two parameters u and m. However, it seems that there is an error, since y is not defined inside the function. Probably, y should be replaced by u or another parameter.

3. Function to calculate the integral numerically:

4. integral[x_, m_] := NIntegrate[dn[y/x, m], {y, -x, x}]

This integral function calculates the numerical integral of the function dn with respect to y, ranging from -x to x. The value of m is passed as a parameter to dn. 5. Function to find solutions:

6. findSolutions[maxPrime_] :=

7. Module[{primes = Prime[Range[maxPrime]], solutions = {}},

8. Do[

9. If[integral[x, m] == integral[Prime[x], m],

10. AppendTo[solutions, {x, Prime[x]}]

11.],

12. {x, maxPrime}

13.];

14. solutions

15.]

This findSolutions function searches for solutions where the integral of x is equal to the integral of the x-th prime number. It:

o Generates a list of prime numbers up to maxPrime.

o Iterates over each number from 1 to maxPrime.

o Compares the integral of x with the integral of the x-th prime.

o If they are equal, adds the pair {x, Prime[x]} to the list of solutions.

16. Usage example:

17. m = 0.5; (* Function parameter dn *)

18. maxPrime = 100; (* Search for solutions up to the 100th prime *)

19. solutions = findSolutions[maxPrime]

Here, the parameter m is set to 0.5 and the function findSolutions is called to search for solutions up to the 100th prime. The result is stored in the variable solutions.

In summary, the code is trying to find values of x for which the integral of the function dn is equal to the integral of the function dn applied to the x-th prime number. However, there is an error in the definition of the function dn that needs to be corrected for the code to work correctly.

What could be an error is actually a random hit if other variables are used to replace y in dn there is no solution to the equation .

```
(* Definindo uma lista de limites *)
limitesN = {10000, 20000, 300000};
limitesY = {104729, 224737, 4256233};

(* Função para calcular os comprimentos de arco e a solução *)
calcularComprimentoECondicao[limiteN_, limiteY_] := Module[
 {arcLength1, arcLength2, condicao, solucao},
 arcLength1 = NIntegrate[Sqrt[1 + D[f1[n], n]^2], {n, -limiteN, limiteN}];
 arcLength2 = NIntegrate[Sqrt[1 + D[f2[y], y]^2], {y, -limiteY, limiteY}];
 condicao = arcLength1 * y / n == arcLength2;
 solucao = FindInstance[condicao, {y, n}];
 {limiteN, limiteY, arcLength1, arcLength2, solucao}
]

(* Criando a tabela de valores *)
tabela = Table[
 calcularComprimentoECondicao[limiteN, limiteY],
 {limiteN, limitesN},
 {limiteY, limitesY}
]

(* Exibindo a tabela *)
Tabela
```

```
{{{10000,104729,131941.,1.3818*10^6,{{y->10.4729,n-
>1}}},{10000,224737,131941.,2.96519*10^6,{{y->22.4737,n-
>1}}},{10000,4256233,131941.,5.6157*10^7,{{y->425.623,n-
>1}}}},{{20000,104729,263881.,1.3818*10^6,{{y->5.23645,n-
>1}}},{20000,224737,263881.,2.96519*10^6,{{y->11.2369,n-
>1}}},{20000,4256233,263881.,5.6157*10^7,{{y->212.812,n-
>1}}}},{{300000,104729,3.95822*10^6,1.3818*10^6,{{y->0.349097,n-
```

```
>1}}},{300000,224737,3.95822*10^6,2.96519*10^6,{{y->0.749123,n->1}}},{300000,4256233,3.95822*10^6,5.6157*10^7,{{y->14.1874,n->1}}}}}
```

```
{{{10000,104729,131941.,1.3818*10^6,{{y->10.4729,n->1}}},{10000,224737,131941.,2.96519*10^6,{{y->22.4737,n->1}}},{10000,4256233,131941.,5.6157*10^7,{{y->425.623,n->1}}}},{{20000,104729,263881.,1.3818*10^6,{{y->5.23645,n->1}}},{20000,224737,263881.,2.96519*10^6,{{y->11.2369,n->1}}},{20000,4256233,263881.,5.6157*10^7,{{y->212.812,n->1}}}},{{300000,104729,3.95822*10^6,1.3818*10^6,{{y->0.349097,n->1}}},{300000,224737,3.95822*10^6,2.96519*10^6,{{y->0.749123,n->1}}},{300000,4256233,3.95822*10^6,5.6157*10^7,{{y->14.1874,n->1}}}}}
```

```
(* Definindo a função para n *)

f1[n_] := n * 0.98844448495 / 0.15158331094


(* Definindo a função para y *)

f2[y_] := y * 0.98844448495 / 0.15158331094


(* Definindo as variáveis para os limites de integração *)

limiteN = 10000;

limiteY = 104729;


(* Calculando a integral definida do comprimento de arco para n *)

arcLength1 = NIntegrate[Sqrt[1 + D[f1[n], n]^2], {n, -limiteN, limiteN}]


(* Calculando a integral definida do comprimento de arco para y *)

arcLength2 = NIntegrate[Sqrt[1 + D[f2[y], y]^2], {y, -limiteY, limiteY}]


(* Definindo a condição *)

condicao = arcLength1 * y / n == arcLength2


(* Encontrando os valores das variáveis que satisfazem a condição *)

solucao = FindInstance[condicao, {y, n}]
```

```
(* Exibindo os resultados *)
{arcLength1, arcLength2, solucao}


131941.
1.3818*10^6
(131941. y)/n==1.3818*10^6
{{y->10.4729,n->1}}
{131941.,1.3818*10^6,{{y->10.4729,n->1}}}
```