**UNIVERSIDADE FEDERAL DE MINAS GERAIS**

# TRABALHO PRÁTICO – MATEMÁTICA DISCRETA

## ESPIRAL QUADRADA

Trabalho apresentado ao professor Antonio Alfredo Ferreira Loureiro, da disciplina de Matemática Discreta, pelo aluno do curso de graduação em Bacharelado em Ciência da Computação, Enok Januário da Rocha como parte das exigências da disciplina.

Minas Gerais

Belo Horizonte, 04 de novembro de 2016

## O PROBLEMA PROPOSTO

O problema proposto consiste basicamente na construção de um programa que fornecerá as coordenadas de um ponto que se encontra disposto em uma espiral quadrada:

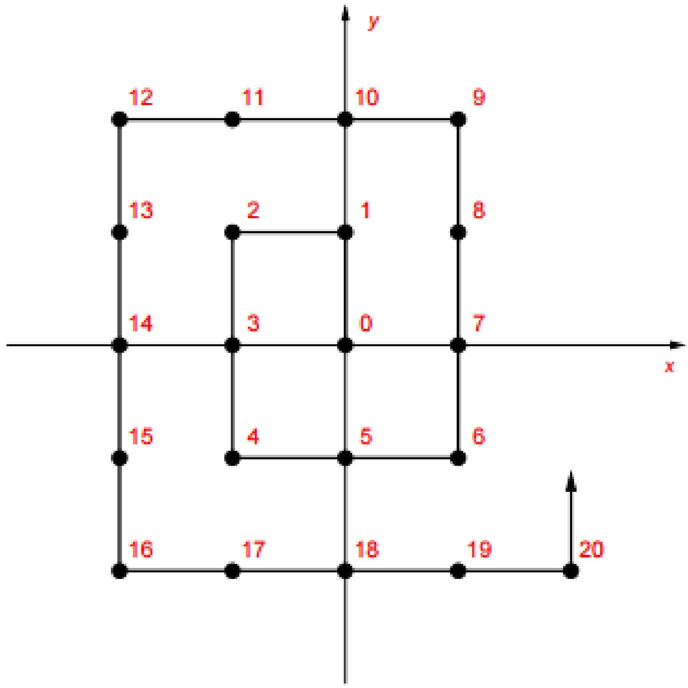


***Figura 1:*** *Esquema mostrando a Espiral Quadrada até o ponto número 20. O esquema foi retirado do PDF referente ao trabalho do professor Antonio Alfredo Ferreira Loureiro.*

A entrada é composta por um inteiro *n,* sendo *n≥0*, que representa o ponto que o usuário deseja acessar. A saída é composta por dois valores que representa as coordenadas (do ponto fornecido na entrada), no formato *(x,y).* As seguintes soluções apresentadas foram testadas em diferentes máquinas com diferentes sistemas operacionais (Windows 10 Pro, GNU/Linux-Ubuntu e Mac OS). Ambas foram implementadas em uma máquina com Windows 10 Pro utilizando o DEV C++.

## PRIMEIRA SOLUÇÃO APRESENTADA

**Complexidade:** *O(n)*

A solução para este problema consiste basicamente em percorrer a espiral, a partir da origem, até o ponto que o programa receber como entrada. Ao analisarmos a espiral, podemos encontrar diversos padrões que facilitam percorrer a espiral quadrada, entretanto, um em especial é o suficiente para realizar tal tarefa com êxito.

A origem se dá pelo ponto 0 que possui coordenada (0,0). A partir desse ponto, é possível perceber que a espiral segue a seguinte ordem de direções para cada passo:

Cima » Esquerda » Baixo » Direita

Em outras palavras, saindo da origem e tentando chegar no próximo ponto, sempre teremos que primeiramente subir, virar à esquerda, descer e posteriormente, virar à direita. Após chegar no último passo, deve-se repetir os procedimentos quantas vezes for necessário, até chegar no ponto desejado. Por outro lado, temos que levar em consideração também, que cada um dos passos é acompanhado de um fator escalar que é comum a um outro passo, isto é, se, por exemplo, é solicitado chegar no ponto de número 6, sabemos que, a partir da origem, teremos que efetuar um passo em direção para cima, um passo para a esquerda, dois passos para baixo e dois passos para a direita. O mesmo fato ocorre para os demais pontos. O que realmente muda de um ponto para outro (mais distante da origem) é quantidade de passos para determinada direção. Sabe-se que a quantidade de passos para cima é igual a quantidade de passos para a esquerda e a quantidade de passos para baixo é a mesma que para a direita, isso se a chegada no ponto desejado não acontecer até a tomada do próximo passo.

O fator que acompanha a quantidade de passos para cima e, consequentemente para a esquerda (CONTADOR_DE_PASSOS1), começa com o valor 1 (em outras palavras, partindo a origem, com ponto objetivo sendo maior que 0 nesse caso em específico, o programa dará ao menos um passo para cima e um passo para a esquerda). Por outro lado, o fator que acompanha a quantidade de passos para baixo e para a direita (CONTADOR_DEPASSOS2), começa valendo 2. Após uma sequência de passos (cima, esquerda, baixo, direita), cada fator aumenta em duas unidades, seus valores. O programa segue essa linha de raciocínio até que o "PONTO_ATUAL" (ponto em que se situa o caminho percorrido no momento), seja igual ou maior ao "PONTO_OBJETIVO" (ponto que o usuário deseja saber a coordenada). Caso o ponto em que o programa se encontra (PONTO_ATUAL), seja maior que o ponto desejado, ele irá desfazer os últimos passos dados, até que se encontre na posição correta. Conforme o trajeto vai sendo realizado seguindo o algoritmo, os valores de X e Y vão sendo alterados. Isto é:

- Caso o movimento tenha direção para cima, o valor de y aumentará proporcionalmente em relação ao CONTADOR_DE_PASSOS1;
- Caso o movimento tenha direção para a esquerda, o valor de x diminuirá proporcionalmente em relação ao CONTADOR_DE_PASSOS1;
- Caso o movimento tenha direção para baixo, o valor de y diminuirá proporcionalmente em relação ao CONTADOR_DE_PASSOS2;
- Caso o movimento tenha direção para a direita, o valor de x aumentará proporcionalmente em relação ao CONTADOR_DE_PASSOS2.

Como dito anteriormente, pode ser que o programa ultrapasse o ponto que busca e então ele terá que desfazer alguns dos passos feitos. Para realizar tal ação, ele poderá

- Subtrair de y, o valor da subtração entre o valor do ponto desejado e o que ele se encontra (caso tenha que deslocar-se para baixo na espiral para atingir seu objetivo);

- Somar a x, a diferença entre o valor do ponto atual e o ponto objetivo (caso tenha que deslocar-se para a direita na espiral para atingir seu objetivo);
- Somar a y, a diferença entre o ponto atual e o ponto objetivo (caso seja necessário deslocar-se para cima para atingir o ponto objetivo).
- Subtrair de x, a diferença entre o ponto atual e o objetivo (caso seja necessário deslocar-se para a esquerda para atingir o ponto objetivo).

O programa para de rodar exatamente quando o PONTO_ATUAL possui o mesmo valor que PONTO_OBJETIVO. Para isso ocorrer, houve a utilização do comando "break" que realiza a parada de execução do loop em que o programa se encontra. Ao final do programa é exibido na tela, as coordenadas do ponto desejado.

**SEGUNDA SOLUÇÃO APRESENTADA:**

**Complexidade:** O $(n^{1/2})$

Analisando a espiral podemos perceber a existência de outros padrões como o utilizado na presente solução. Os quadrados perfeitos pares se encontram dispostos no terceiro quadrante alinhados (ver Figura 2) e para encontrar a coordenada de um deles, basta dividir sua raiz quadrada por dois e inverter seu sinal. Exemplo: Tomando o ponto 4 temos que:

- A raiz quadrada será 2;
- A raiz quadrada dividida por 2 resultará em 1.
- Invertendo o sinal deste valor, temos -1. Logo, as coordenadas do ponto 4 são (-1,-1).

Outro fato interessante, utilizado na construção desta solução, é que os quadrados perfeitos ímpares se encontram dispostos no primeiro quadrante alinhados (ver Figura 2) e para encontrar a coordenada de um deles, basta seguir o seguinte processo:

- O valor de x será dado pelo piso da raiz quadrada do ponto (sendo este um quadrado perfeito ímpar) dividida por 2.

$$X=\lfloor sqrt(ponto)/2 \rfloor$$

- O valor de y será dado pelo teto da raiz quadrada do ponto (sendo este um quadrado perfeito ímpar) dividida por 2.

$$Y=\lceil sqrt(ponto)/2 \rceil$$

Sabendo de tais fatos, fica um pouco mais fácil efetuar a construção de um programa capaz de encontrar a coordenada de qualquer ponto pertencente a espiral quadrada. Para isto, primeiramente deve-se identificar qual é o quadrado perfeito mais próximo do ponto digitado. O ponto mais próximo será encontrado, a partir da multiplicação da raiz inteira do ponto digitado por ela mesma, em outras palavras, deve-se arredondar a raiz do número obtido na entrada e elevá-la ao quadrado.

Feito isso, chegamos a um novo problema: descobrir se o quadrado perfeito mais próximo é par ou ímpar. Isso é feito por 2 condicionais principais que comparam o resto da divisão do PONTO_PROXIMO (quadrado perfeito mais próximo do ponto desejado).

**PRIMEIRA CONDICIONAL PRINCIPAL -** Caso PONTO_PROXIMO seja um ímpar, então sabemos que o quadrado perfeito mais próximo é um ímpar e teremos duas possibilidades:

- O PONTO_PROXIMO ser maior que o ponto desejado (ponto dado como entrada no início do problema). Como o ponto que buscamos, se encontra próximo a um quadrado perfeito ímpar e ele está abaixo dele, inferimos que ambos os pontos terão o mesmo valor para a coordenada x e devemos calcular a distância entre tais para descobrirmos o calor de y para o ponto dado como entrada. Neste caso, o valor para x, da coordenada do ponto desejado, será o piso da divisão da raiz quadrada do PONTO_PROXIMO por 2, e o y será o teto da divisão entre o PONTO_PROXIMO por 2, subtraindo a diferença entre o ponto desejado e o PONTO_PROXIMO. Matematicamente:

X=⌊sqrt(PONTO_PROXIMO)/2⌋

Y=⌈sqrt(PONTO_PROXIMO)/2⌉-(ponto-PONTO_PROXIMO)*,* sendo “ponto”, o ponto dado pelo usuário do programa.

- O PONTO_PROXIMO ser menor que o ponto desejado. Como o ponto que buscamos se encontro próximo a um quadrado perfeito ímpar e ele está ao lado esquerdo deste quadrado perfeito, devemos manter o valor de y da coordenada do quadrado perfeito e devemos calcular o x de acordo com a distância entre tais pontos. Matematicamente:

X=⌊sqrt(PONTO_PROXIMO)/2⌋-(ponto-PONTO_PROXIMO)
Y=⌈sqrt(PONTO_PROXIMO)/2⌉

**SEGUNDA CONDICIONAL PRINCIPAL-** Caso PONTO_PROXIMO seja par, então é trivial que se trata de um quadrado perfeito par. Então:

- Se o PONTO_PROXIMO for maior que o ponto que desejamos saber a coordenada, então sabemos que o ponto desejado estará acima do PONTO_PROXIMO e, consequentemente possuirá o mesmo valor de x que o quadrado perfeito par próximo (PONTO_PROXIMO). Já o valor de y será dado pela raiz quadrada do PONTO_PROXIMO dividida por 2 e subtraída pela distância entre os dois pontos. Como estamos tratando de pontos que são se encontram nos quadrantes com sinal negativo, devemos inverter os sinais das coordenadas encontradas. Matematicamente:

X= - (sqrt(PONTO_PROXIMO)/2)
Y= - [(sqrt(PONTO_PROXIMO)/2-(ponto-PONTO_PROXIMO))]

- Se o PONTO_PROXIMO for menor que o ponto que desejamos saber a coordenada, então sabemos que este ponto estará a direita do PONTO_PROXIMO e consequentemente possuíra o mesmo valor de y que ele e seu x será dado pela raiz quadrada do PONTO_PROXIMO dividida por dois e acrescida pelo valor da distância entre os dois pontos tratado. Lembrando sempre que, por estarmos tratando de pontos nos quadrantes negativos, devemos inverter o sinal da computação efetuada. Matematicamente:

X= - [(sqrt(PONTO_PROXIMO)) / 2+(PONTO_PROXIMO-ponto)]
Y= - [(sqrt(PONTO_PROXIMO))/2]

**TERCEIRA CONDICIONAL PRINCIPAL-** Há também a possibilidade do usuário deseja saber a coordenada de um número que é quadrado perfeito, seja ele par ou ímpar. Nesse caso a variável "ponto" será igual a "PONTO_PROXIMO" e assim, será executada a parte que se encontra no final do código:

- Caso a entrada seja um quadrado perfeito ímpar, temos que seu x será igual ao piso da sua raiz quadrada dividida por 2 e seu y será dado pelo teto da raiz quadrada do ponto dividida por 2.

$$X=\lfloor((\mathrm{sqrt}(ponto))/2)\rfloor$$

$$Y=\lceil((\mathrm{sqrt}(ponto))/2)\rceil$$

- Caso a entrada seja um quadrado perfeito par, temos que seu valor de x será idêntico ao seu valor de y e ambos serão dados por:

$$X=Y=((\mathrm{sqrt}(ponto))/2)$$

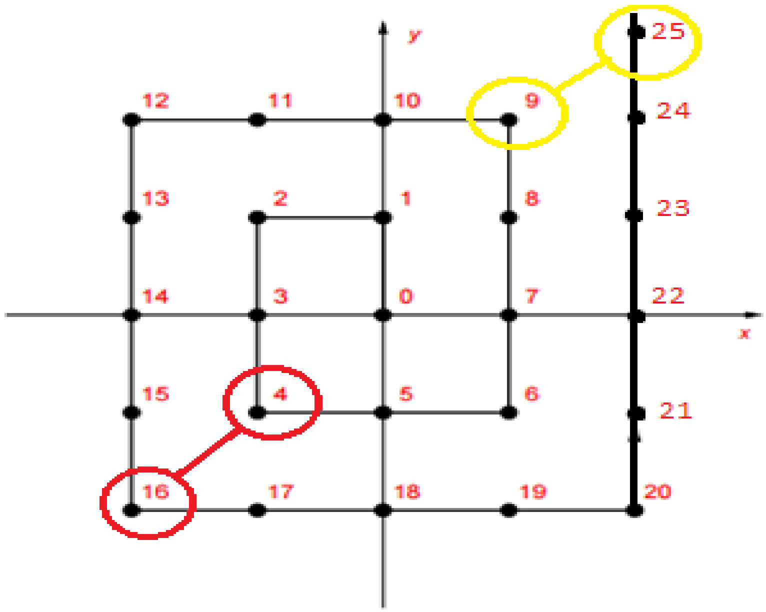


*__Figura 2:__ Esquema mostrando a Espiral Quadrada até o ponto número 25. Os círculos amarelos indicam os quadrados perfeitos ímpares e os círculos vermelhos indicam os*

*quadrados perfeitos pares. O esquema foi retirado do PDF referente ao trabalho do professor Antonio Alfredo Ferreira Loureiro e editado pelo autor deste trabalho.*

Uma informação que não pode deixar de ser feita é que, para a construção dessa solução, foram utilizadas as funções *sqrt, ceil e floor* que encontram-se disponíveis na biblioteca *math.h.*

**CONSIDERAÇÕES FINAIS**

Este trabalho foi apresentado ao professor Antonio Alfredo Ferreira Loureiro, da disciplina de Matemática Discreta, e contém duas soluções para o problema proposto. A entrada e saída de ambas as soluções são a mesma, isto é, em relação a forma como são efetuadas. Juntamente com este documento, estão sendo entregues dois arquivos fonte: espiral.c, espiral2.c e um arquivo texto leiame.txt com as instruções de execução de cada solução enviada pelo aluno.