TRABALHO PRATICO 1
Matemática discreta
UFMG-2021/1

Mateus Henrique Souza Silva

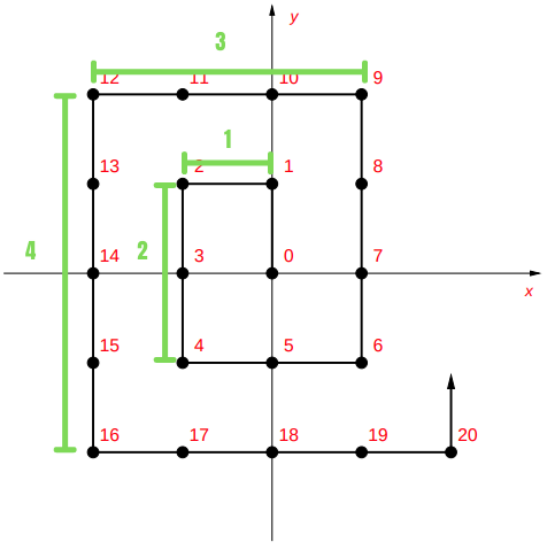
Espiral quadrada:

Para a resolução do problema da espiral quadrada, primeiro foram observados alguns padrões:

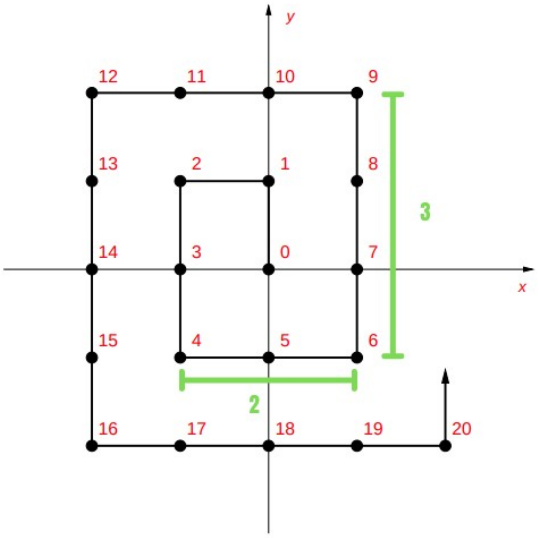
1. Alternância na quantidade de “saltos” que a espiral realiza, de acordo com a alternância de direção em que ela vai.

   Ex:

   Sempre que a direção da espiral se altera da esquerda, para baixo, a quantidade de saltos aumenta em 1, como pode ser observado a seguir:



   O mesmo acontece, quando a direção se altera da direita para cima:



2. O valor alterado de acordo com cada direção, ou seja, quando a espiral está direcionada para cima, o valor a ser incrementado é Y, deixando X estático. Já quando direcionada para a esquerda, o valor de X deve ser decrementado. Para baixo, o valor a ser decrementado é Y. E por fim, quando direcionada para a direita, o valor de X deve ser incrementado.

Observados esse padrões, foi possível resolver o problema de forma simples.

Para representar as direções da espiral, foi usada uma struct que guardava sua letra (c: cima, d: direita, e: esquerda e b: baixo), e qual era a próxima direção na ordem seguida pela espiral.

```
typedef struct Dir{
  char letter;
  Dir* next;
}Dir;
```

Ao iniciar o programa, o usuário informa o numero do ponto o qual deseja-se descobrir as coordenadas. Após isso, é feita uma iteração com um ‘for’ simulando a criação da espiral de acordo com os padrões observados, calculando-se os valores de x e y para cada ponto até que se chegue ao ponto informado pelo usuário.

Ao final, temos um algoritmo de complexidade O(n), devido ao fato de iterar apenas um laço ‘for’.

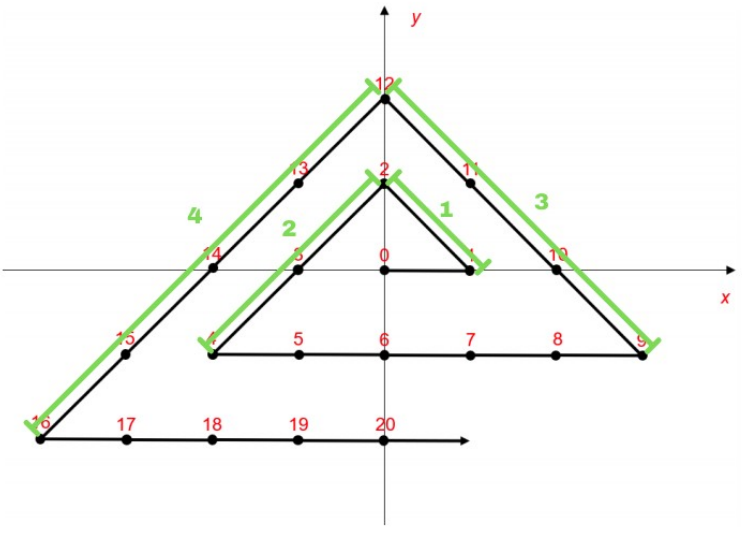
Espiral triangular

O problema da espiral triangular foi resolvido de forma semelhante à espiral quadrada, observando-se os mesmo padrões, porém com algumas particularidades:

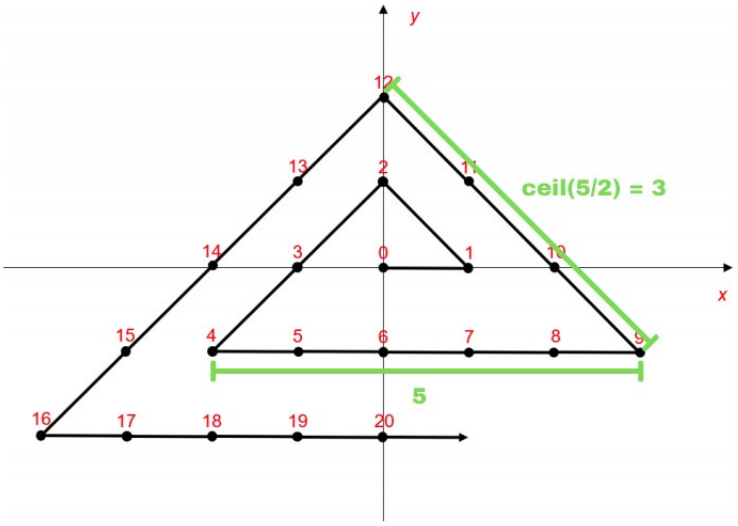
1. Assim como na espiral quadrada, a alternância na quantidade de “saltos” de acordo com a direção, também se faz presente, porém com algumas diferenças:

   Ex:

   Sempre que a direção da espiral se altera da diagonal esquerda superior para a diagonal esquerda inferior, o numero de passos aumenta em 1:



Já quando a direção se altera da direita para a esquerda, a quantidade de passos varia de x para ⌈x/2⌉, como pode ser visto a seguir:



2. O valor alterado de acordo com cada direção, ou seja, quando a espiral está direcionada para diagonal esquerda superior, o valor de Y é incrementado e o de X é decrementado. Já quando direcionada para a diagonal esquerda inferior o valor de X e de Y devem ser decrementados. E por fim, quando direcionada para a direita, o valor de X deve ser incrementado e deixar o valor de Y estático.

Utilizando-se de uma estrutura de dados igual a do problema da espiral quadrada, guardamos a direção e a próxima, na ordem da espiral, que no caso é direita → cima → esquerda → direita… e assim sucessivamente.

```
typedef struct Dir{
  char letter;
  Dir* next;
}Dir;
```

Ao iniciar o programa, o usuário informa o número do ponto o qual deseja saber as coordenadas, e o programa simula a criação da espiral até o ponto desejado por meio de um laço ‘for’, iterando sobre as direções e calculando x e y de acordo com cada ponto. Ao final, temos um algoritmo O(n) devido ao fato de o laço ‘for’ iterar sempre até o número informado pelo usuário.