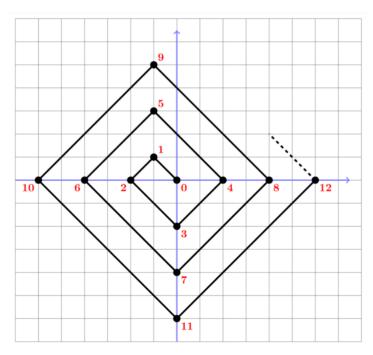
Caracol Noroeste

Gabriela Tavares Barreto

Matrícula: 2018074657

Introdução

Como trabalho prático da disciplina de Matemática Discreta, no semestre de 2022/1, foi proposto criar uma solução para o cálculo das coordenadas do caracol noroeste. Neste trabalho foi utilizada



uma estratégia com custo $\Theta(1)$. Essa solução se dá da seguinte maneira:

 Considerando o lado direito do caracol, vemos que todos os pontos estão no eixo das ordenas,, e que são múltiplos de 4. Isso indica que, para encontrarmos um ponto **n** qualquer do caracol, basta acharmos qual é o múltiplo de 4 que seja menor ou igual à n. Uma vez identificado esse múltiplo de 4, em apenas um passo chega-se ao par de coordenadas desejadas. Assim, o custo computacional em termos da quantidade de coordenadas calculadas está limitado pela constante 1. Após uma observação do comportamento do caracol, é possível perceber padrões para os valores das

coordenadas dependendo do eixo em que ela está e do valor do ponto n. Dessa forma foi possível estabelecer os passos para o cálculo de cada ponto n, após encontrar o múltiplo **m** de 4 procurado:

a. Se n = m: todos os múltiplos de 4 encontram-se no eixo x, do lado positivo. Logo para eles y sempre será igual a zero, e x será o múltiplo m dividido por 2.

$$(x, y) = (m/2, 0)$$

b. **Se n = m+1:** todos os pontos iguais a um múltiplo de quatro somado a um estão no "eixo norte" do caracol, distanciados por um valor -1(no eixo x) em relação à origem. Logo para esses pontos, x sempre será igual a -1, e y será (m/2) + 1.

$$(x, y) = (-1, (m/2)+1)$$

c. Se n = m+2: todos esses pontos estão localizados no lado negativo da reta do eixo x. Logo para eles y será sempre zero, e x valerá (-m/2) -2.

$$(x, y) = (-(m/2) - 2, 0)$$

d. Se n = m+3: esses pontos se encontram no "eixo sul" do caracol, justamente sobre a parte negativa da linha das ordenadas. Sendo assim, eles tem x sempre igual a zero, e y será (-m/2) -2.

$$(x, y) = (0, -(m/2)-2)$$

Assim, começamos calculando m, e calculando a diferença de n-m. A partir dessa diferença, saberemos qual dos quatro casos acima apresentados deveremos aplicar a n!

O Programa

Estruturas

Para esse programa foi definida uma estrutura do tipo ponto, que armazena três inteiros: as coordenadas (x, y) e o valor n do ponto.

```
typedef struct Ponto{
   int x, y, n;
}Ponto;
```

Funções

Foram criadas duas funções, a menor_multiplo_de_4 e calcula_ponto.

1- menor_multiplo_de_4

Essa função recebe um inteiro x, e retorna o inteiro múltiplo de 4 menor que ele, que é também o mais próximo a x.

```
int menor_multiplo_de_4(int x){
   int quociente = x/4;
   return quociente*4;
}
```

2- calcula_ponto

Já a função calcula_ponto, recebe um ponteiro para uma variável do tipo Ponto.

```
void calcula_ponto(Ponto *p){
```

Ao chamar a função, é declarada uma variável do tipo chama **ponto_ref**, que servirá como referência para os cálculos a serem feitos. A ela é atribuído o valor de menor_multiplo_de_4 do valor n do ponto passado. Depois disso, é calculada a diferença entre n de p(ponto desejado) e ponto_ref, que é armazenado na variável dif.

```
int ponto_ref;
ponto_ref = menor_multiplo_de_4(p->n);
int dif = p->n-ponto_ref;
```

Depois disso é feito um condicional do tipo switch. Em cada **case** será atribuído os valores corretos das variáveis x e y de p. De acordo com a estratégia já apresentada na introdução deste trabalho.

```
case 2:
switch (dif){
                                              p->x = -ponto_ref/2 - 2;
case 0:
                                              p \rightarrow y = 0;
    p->x = ponto_ref/2;
                                              break;
    p \rightarrow y = 0;
    break;
                                        case 3:
                                              p->x = 0;
case 1:
                                              p-y = -ponto_ref/2 - 2;
    p \rightarrow x = -1;
    p->y = ponto_ref/2 + 1;
                                              break;
    break;
                                         return;
```

Após a atribuição de valores, é forçada a saída do switch e a função termina.

main

Por fim, é criada a função principal main, na qual é criada uma variável ponto do tipo Ponto. Ela

```
void main(){
   Ponto *ponto;
   scanf("%d", &(ponto->n));

   ponto->x=0;
   ponto->y=0;

   calcula_ponto(ponto);
   printf("%d %d", ponto->x, ponto->y);

   return;
}
```

recebe um valor para n, digitado pelo usuário, por meio da função scanf. Depois suas coordenadas x e y são zeradas. Após isso, é chamada a função calcula_ponto. Depois disso, pela função print é feita a saída do programa, com as respectivas variáveis x e y atribuídas ao ponto inserido pelo usuário, e o programa é encerrado.