

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

- 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.

```
switch (dif){
case 0:
    p->x = ponto_ref/2;
    p->y = 0;
    break;

case 1:
    p->x = -1;
    p->y = ponto_ref/2 + 1;
    break;
```

```
case 2:
    p->x = -ponto_ref/2 - 2;
    p->y = 0;
    break;

case 3:
    p->x = 0;
    p->y = -ponto_ref/2 - 2;
    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.