TRABALHO 02

01. Defina uma constante/macrode valor 10 associado a dimensão da matriz 10x10;
02. Defina uma constante/macrode valor 100 associado ao tamanho máximo da minhoca;
03. Defina estruturas (sugestões de estruturas):
- Estrutura que representa a posição e possui dois elementos do tipo inteiro: x e y. Essa estrutura representa uma posição na matriz;
- Estrutura que representa a minhoca que contém 2 elementos: um inteiro com o tamanho atual da minhoca e um ponteiro para estrutura posição que representa um vetor das posições da minhoca;
- Estrutura que representa o doce que possui dois elementos: uma estrutura posição e um inteiro vida. Essa estrutura possui a sua posição na matriz e o vida representa: durante quantas jogadas o doce ficará nessa posição.

04. Defina variáveis globais (sugestões de variáveis globais):
- Uma variável global que representa a minhoca;
- Uma variável que representa o doce;
- Uma variável global que representa a matriz do jogo.

05. O programa deve conter obrigatoriamente, NO MÍNIMO, as seguintes funções com no mínimo os parâmetros informados (você poderá adicionar mais parâmetros assim como criar outras funções)
- gerarDoce(): muda a posição do doce na matriz. A nova posição deve ser aleatória;
- movimentar(): movimenta a minhoca para uma nova posição
- ponteiroNulo(v) que retorna falso se a alocação/realocação aconteceu com sucesso se não o programa deve ser encerrado

06. O programa deve movimentar a minhoca para cima (tecla W), para baixo (tecla S), esquerda (tecla A) e direita (tecla D).
07. Considere as seguintes restrições:
- A minhoca deve se movimentar sozinha na direção atual até que mude a direção dela;
- É movimentada através das teclas e quando um doce aparece em algum lugar da matriz a vida do doce é definida pela quantidade de movimentações que a minhoca fará. Suponha que a vida do doce é 4, significa que se a minhoca não chegar na posição do doce em 4 movimentações, o doce deve sumir e aparecer em outro local.
- Quando a minhoca chegar na posição do doce, o seu tamanho que o ponteiro para estrutura posição (vetor de posições) deve aumentar mais um elemento através da realocação dinâmica de memória. O doce deve então aparecer em outra posição aleatoria (essa posição aleatoria pode ser também a posição que a minhoca está, mas o doce só será consumido se a cabeça da minhoca ocupar a posição do doce)
- A minhoca começa a partida com o seu tamanho = 3, ou seja, o elemento ponteiro para estrutura posição (vetor de posições) deve ser alocado dinamicamente com 3 posições. A minhoca deve iniciar a partida com tamanho 3 nas posições x e y de cada posição em (0,0) (0,1) (0,2)
- Se a minhoca esbarrar em alguma parte do seu corpo, mostrar a mensagem "Game Over!" , a região do ponteiro de posição da minhoca deve ser liberado e o jogo deve ser encerrado.
- Sobre os limites do jogo, você pode optar por deixar que a minhoca atravesse a parede OU definir que se ela esbarrar na parede acontece o apresentado na restrição anterior.
- Se o jogador teclar a tecla Q, o jogo deve ser encerrado.
- Se o jogador teclar qualquer outra tecla, deve ser informado quais teclas são permitidas
- A tecla pressionada pelo jogador não deve ser exibida na tela (utilize as soluções encontradas pelos colegas no trabalho anterior)
- Utilize o comando switch-case
- Para limpar o terminal a cada novo movimento utilize : system("@cls||clear");
- Para gerar um número aleatório entre 0 e 10 utilize o trecho:

```
#include <time.h>
#include <stdlib.h>
#include <stdio.h>

int main(void)
{
  int numero_randomico;
  srand((unsigned) time(NULL));
  numero_randomico = rand()%10;
  printf("%d", numero_randomico);
}
```

Conhecimentos que DEVEM ser aplicados:
- **Constantes ou macros**
- **Variáveis globais**
- **Funções**
- **Matrizes**
- **Estruturas**
- **Ponteiros, alocação e realocação de memória e liberação de memória**