Atividade de Revisão

Introdução

- **Ponteiro:**
  - "Tipo de dado cujo valor se refere a um outro valor alocado em outra área da memória"
  - Utilizamos o simbolo * para "converter" uma variável para ponteiro, ou "vice-versa".
  - Com relação a ponteiros existem alguns termos:
    - referência
    - desreferênciar
    - alocar
    - desalocar
    - apontar

  - **Analogia:** um ponteiro é como um link, ou seja, eu tenho o caminho padrão para fazer um arquivo, porém o ponteiro é um outro caminho que tenho para o mesmo arquivo.
  - **Objetivo:** Quando passamos uma variável como argumento de uma função esta é passada por cópia, logo não podemos alterar o seu valor durante o processo. Por isso, ponteiro serve para trabalhar com posição da memória para alterar e trabalhar com os valores reais.
  - **Alocação Dinâmica:** Tem como objetivo reservar uma região da memória para armazenar dados. Geralmente é utilizado quando usamos uma função para criar um objeto, pois, uma variável criada dentro de uma função so existe dentro do seu escopo.

```
// Ponteiro
int x = 10;  // variável que carrega o valor 10
    // esta variável pode tras informações: &x -  endereço da memória de x
int* y; //ponteiro y do tipo inteiro
y = &x; // com isso temos acesso ao valor de x através do próprio e do y.

//podemos acessar o valor de x, através de y, da seguinte forma
std::cout << *y ; //desrefênciando y, ou seja, ao invés de printar a posição na memória, te
```

- **Estrutura:**
  - Da forma mais abstrata possível, uma estrutura (*struct*), é uma "variável" que você define, ou seja, é um novo tipo de dado (apesar do tipo ser Struct), onde você pode fazer uma variável composta, uma variável que trás dentro de si outras informações.
  - É, basicamente, um bloco de código onde definimos variáveis que estarão associadas a estrutura que estamos criando.

```
// struct
```

```
struct Aluno{
    string nome;
    int idade;
    int matricula;
};
```

- **Recursão:**
  - "É um método de resolução de problemas que envolve quebrar um problema em subproblemas menores e menores até chegar a um problema pequeno o suficiente para que ele possa ser resolvido trivialmente".
    - Uma forma de resolução de problemas utilizando o conceito de chamar uma função dentro dela mesma, onde nos atentamos ao **caso base** e o **chamada da recursão**.
  - –

```
//Recursão
  // forma recursiva do fibonacci 1 1 2 3 5 ...  , onde x é a posição na sequencia
int fib(int x){
  if (x==0) return 0 // casos base 1
  else if(x == 1 || x == 2) return 1 // caso base 2
  else
      return fib(x-1)+fib(x-2)  // como pode ver temos a chamada da recursão e o(s) cas
}
```

Proposta

Vamos fazer um joguinho...

Referência

- panda