1 - SOBRE PONTEIROS

1 - Vamos observar um codigo

```c
int main(){//inicio_main

    printf("\n");

    int contador = 10;

    printf("Antes de incrementar.\n", contador);
    printf("O contador vale :%d\n",contador);
    chamada de função 1
    incrementa(contador); //passa a variavel contador para a função incrementa.

    printf("Depois de incrementar.\n", contador);
    printf("O contador vale :%d\n", contador);

    //declarando_variaveis
    //entrada_dados      You, a few seconds ago • Uncommitted changes
    //processamento_dados
    //saida_dados
    printf("\n");
    printf("\n");
    return 0;
}//fin_main
```

```c
////////////////////////////////////////FUNÇÕES/////////////////////////////////////////////////

//1//
void incrementa(int valor){
    printf("O Antes de incrementar.\n");        IMPRIME CONTADOR
    printf("O contador vale :%d\n",valor);

    printf("O Depois de incrementar.\n");
    //valor++;                                  ~2 FORMAS~
    printf("O contador vale:%d\n",++valor);     1 - valor++(incrementa DEPOIS da execução)
    //printf("O contador vale :%d\n",valor);    2 - ++valor(incrementa ANTES da execução)
```

```
Antes de incrementar.
O contador vale :10
O Antes de incrementar.
O contador vale :10
O Depois de incrementar.
O contador vale:11
Depois de incrementar.
O contador vale :10
```

- A QUESTÃƒO Ã‰, PQ DEPOIS DE **INCREMENTAR** AINDA CONTINUA DANDO 10?
- O que queriamos Ã© que quando a funÃ§Ã£o incrementa for chamada, ela adicione 1 ao contador e depois retorne ela incrementada, mas isso nÃ£o esta acontecendo, ela so esta incrementando dentro da execuÃ§Ã£o da funÃ§Ã£o.

- Isso acontece pq quando inicializamos **contador =10;** e passamos **contador** como parametro estamos passando para a funÃ§Ã£o somente o valor 10 = COPIA POR VALOR.
Â Â Â - Quando declaramos uma variavel a linguagem C aloca um espaÃ§o em memoria para colocar este valor.
Â