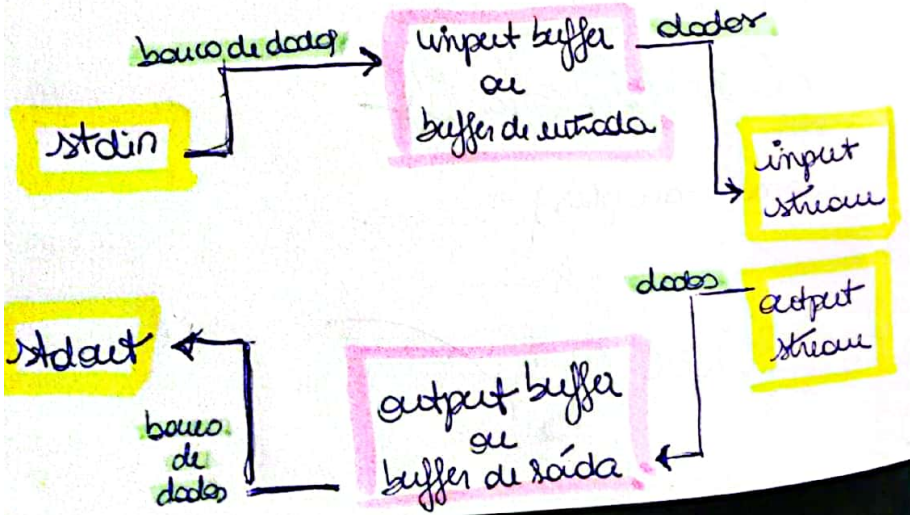
Week 1 - C

O que é buffer?
+ como fazer limpeza de buffer?

- Não existe uma maneira de ~~limpar~~ limpar buffer que seja um consenso entre os programadores, há várias maneiras de fazer isso;

- Funções auxiliam na leitura de dados do programa ou escritura de dados no nosso problema no local



→ Quando eu chamo a função no input, por ex. a função getchar, e aperto o caractere 'A' no teclado, ela fica ali esperando eu apertar a tecla enter.

Esse tempo de espera até apertar a tecla enter, a letra 'A' é armazenada no buffer, e isso é muito importante.

Buffer é um local TEMPORÁRIO na minha memória, que armazena dados até que o buffer encha ou receba um comando que libere os dados armazenados pelo ~~dentro~~ do programa.

(INPUT STREAM)

o "A", ~~depois~~ do enter, irá isso. :)
Enquanto isso, ele está no buffer.

---

enter = \n → Então, no meu buffer, teremos 2 caracteres: A e \n. Só que a função getchar vai ler apenas 1 deles.

1º) lerá o caractere 'A' e jogará pl dentro do input Stream;

2º) o '\n' vai continuar dentro do buffer como uma sujeira, perdida.

↓

Limpeza de buffer precisa ser feita

Porque isso influenciará no meu programa.

```
#include <stdio.h>
int main (void) {
    char a, b, c;

    a = getchar();
    b = getchar();
    c = getchar();

    printf ("%c", a);
    printf ("%c", b);
    printf ("%c", c);

    return 0;
}
```

No output, temos que fornecer a info do programa.

pl o 1º getchar, → (1)

Então, o nº 1 foi jogado pl o caract. (a)

Depois, aperto o nº 2 ↓

```
1
2
1
2
```

← ele automaticamente preenche com 1 e 2. e o programa termina

Como os 3 getchar foram executados (e eu apertei enter apenas pl 2 deles)

- No 1º getchar(a), ficou a sujeira da tecla enter no meu buffer.
- No 2º getchar(b), ele leu no buffer de entrada que já tinha o caractere '\n'. Ele pegou a tecla enter e jogou no caractere 'b'
- No 3º chamada, ficou muito sujeira.

Como resolver?

```
a = getchar();
fflush(stdin);
b = getchar();
fflush(stdin);
c = getchar();
fflush(stdin);
```

isso funciona APENAS pl Windows!

Se formos na descrição da função **fflush** na documentação oficial, ela RECEBE um ponteiro do tipo

FILE

↓

stdin ✓

Agora, na descrição, se o file aponta pl uma string de output, então isso já funciona.

↳ pq é input, não é standard de output!

Ou seja, se a função fflush receber um ponteiro file do tipo output, por ex. stdout, e no buffer tiver qualquer dado que eu tenha escrito no output, a função faz com que qualquer sujeira seja deletada pl o host do sistema operacional escrito no console.

Teoricamente, há 1 lógica por trás disso!

mas jogar stdin vai contra a regra da função, tem que ser do tipo output.

---

② Outra... setbuf(stdin, NULL)

↓ NÃO funciona! (no MAC).

stdin o buffer de entrada ↑ pl nulo.

---

③ Criamos uma própria função...

→

```
void limparBuffer(void){
    char c;
    while((c = getchar()) != '\n' && c != EOF);
}
```

Quando getchar vai no meu buffer e vai ler algo se estiver lá.

e vai ler ...

Vai fazer isso enquanto ele não achar o caractere ('\n') e enquanto não achar o final do arquivo.

→ Se encontrar o enter ('\n'), para de ser executado.

Portanto, sobrescrever é uma ótima opção. 😉

Vamos ver isso ...

```
#include <stdio.h>

int main(void){
    void limparBuffer(void);
    char a,b,c;

    a = getchar();
    limparBuffer()
    b = getchar();
    limparBuffer()
    c = getchar();
    limparBuffer()

    printf("%c",a);
    printf("%c",b);
    printf("%c",c);

    return 0;
}

void limparBuffer(void){
    char c;
    while((c = getchar()) != '\n' && c != EOF);
}
```