Nº 97368	Nome: João Pedro Vilas Boas Braga	Turma: PL5 MiEi

Resolução dos exercícios (deve ser redigido manualmente)

1. Código C de um `main` simples que invoque a função `getline`

Copie para aqui o código C de um `main` simples que colocou no servidor remoto (para invocar a função `getline`).

```
int main() {
    printf("%s\n", getline());
    return 0;
}
```

2. Análise do código desmontado

Compile o código C sem qualquer otimização (com `-O0`) e **copie** para aqui o código executável "desmontado" (*disassembled*) da função `getline` até à chamada da função `gets`, mostrando (com um *print screen* ou foto do monitor) todos os comandos que usou para compilar e para ter o código desmontado da função.
Anote detalhadamente o código desmontado, ignorando as fases de arranque e término da função.

```
push %ebp
mov %esp,%ebp   } Arranque
sub $0x18,%esp  - esp = esp - 0x18
sub $0xc,%esp   - esp = esp - 0xc
lea 0xfffffff8(%ebp),%eax  - eax = ebp-8 (eax é o endereço de buf)
push %eax       - coloca na stack o endereço de buf
call 0x8048328  - invoca gets
```

3. Execução do código

Replique aqui tudo que apareceu no monitor assim que mandou executar o código (incluindo os caracteres que tiver introduzido e o resultado da execução do código).

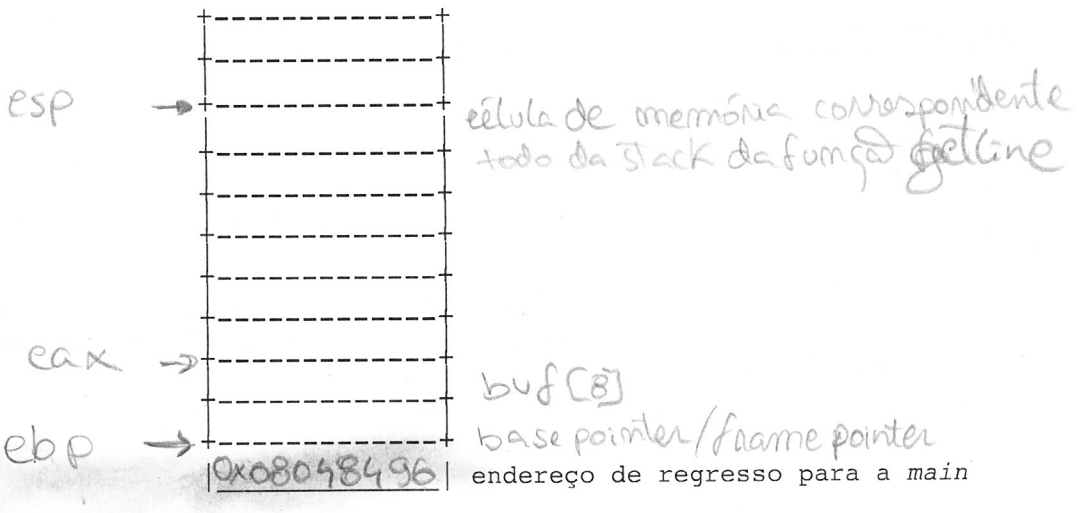
```
$ ./getline
123456789
123456789
Segmentation fault
$
```

4. Estimando o quadro da função `getline` na *stack*

A anomalia que constatou poderá ser devida a um erro na função `getline` que provocou a alteração indevida de informação armazenada na *stack*. Isso pode conduzir a que a execução do código tenha tentado aceder a uma zona de memória que não faz parte da área de memória que estava alocada a este programa. Tal pode acontecer no regresso de uma função, se o valor do endereço de regresso (que está na *stack*) tiver sido indevidamente modificado.
Para verificar se foi isto que aconteceu, temos de analisar o quadro da função `getline`.

Preencha o diagrama do quadro da função `getline` (a sua *stack frame*) que é gerado até este ponto da execução (até à linha 5), com a estimativa dos seus valores e endereços, procedendo assim:

a. **Indique** à esquerda das caixas (cada caixa representa 4 células de memória) a posição apontada pelo registo `%esp` e pelo registo `%ebp`, bem como outras posições relevantes (nesse caso, utilize posições relativas ao registo `%ebp`, e.g., `%ebp+4`, etc.);
b. **coloque** à direita de cada caixa uma etiqueta que descreva o que representa a caixa (nota: algumas caixas podem conter valores irrelevantes);
c. **coloque** dentro da caixa correspondente o endereço de regresso para a `main` (nota: consulte o código desmontado da função `main`)



5. Confirmação de valores do quadro da função `getline` na *stack*

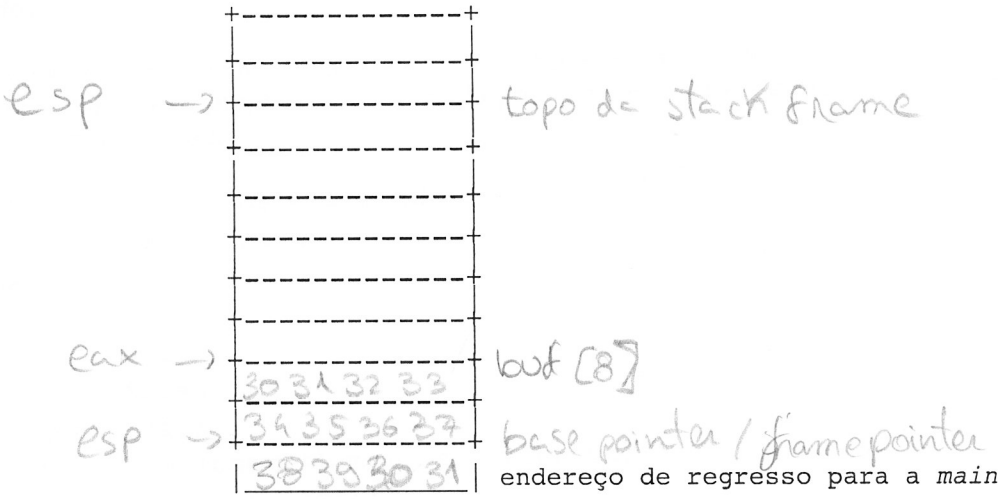
Vamos confirmar a *stack frame* que construiu, colocando um *breakpoint* na linha 5 de `getline` e executando o programa.
Quando este parar no *breakpoint*, veja os valores dos registos relevantes para a *stack frame* (`%ebp` e `%esp`), bem como o conteúdo da posição da *stack* onde está armazenado o endereço de regresso.

Coloque os comandos/resultados que utilizou para obter esses 3 valores (apresente da mesma maneira que apareceu no seu monitor).

```
(gdb) info registers
eax    0xbffffabf0
...
esp    0xbffffabd4
ebp    0xbffffabf8
```

6. Nova análise do quadro da função `getline` na *stack*

Preencha o diagrama seguinte relativo à *stack frame* de `getline`, estimando os valores dos conteúdos das caixas, após a execução da função `gets`, usando a *string* de 12 carateres sugerida no exercício **3**.



7. (e 8.) Explicação da alteração do quadro da função `getline` na *stack*

Identifique no diagrama em cima as células de memória da *stack frame* que foram alteradas após executar a função `gets`.
Descreva detalhadamente o impacto destas alterações na restante execução do programa.
Identifique o(s) registo(s) que foi(oram) corrompido(s) no regresso da função `getline` e mostre como foram modificados.
Identifique e caracterize os problemas associados à utilização da função `gets`.

Foi alterado as células de memória que contêm o endereço de regresso para a main
Com isto, a função, após ser executada, ao obter o endereço de regresso obterá um endereço que não corresponde à operação seguinte, fazendo o programa "crashar".

?

A função gets lê caracteres e coloca-os na stack sem nenhum tipo de controlo, o que pode levar a esta função reescrever células de memória que não deveria.