Programação em Sistemas Computacionais
Segundo Mini-teste na turma: **LI31N**, 16 de Dezembro de 2010

Nome: _____ **Número:** _____ **Turma:** LI31N |A

Considere, em todas as questões, a utilização de uma arquitectura *little endian* a 32 bits.

1. **[4]** Considere os parâmetros para as diferentes *caches*. Preencha os parâmetros vazios.

Cache	m	C	B	E	S	t	s	b
1.	32	1024	32	4	_____	_____	_____	_____
2.	32	_____	8	1	_____	_____	6	_____

Legenda:
- m – número de bits do endereço físico
- B – dimensão do bloco (linha) em bytes
- E – determina a associatividade (número de *ways*)
- s – número de bits do índice do *Set*
- C – dimensão da *cache* em bytes
- S – número de conjuntos (*Set*) da cache
- t – número de bits da *Tag*
- b – número de bits do *offset* no bloco (linha)

1.b **[2]** Considere `sizeof(int) == 4`, a variável global `int dat[N]` e a função `cyclic_read`, que consiste num ciclo infinito de leitura de todas as posições do *array* `dat` e que utiliza apenas os registos do processador para manter as variáveis auxiliares de que necessita. Para a *cache* 1, qual o maior valor de `N` que permite executar os ciclos sem ocorrência de *misses*? _____

2. **[4]** Qual o resultado para o seguinte programa em C

```
#include <stdio.h>
#define ADD_INT(A,B) ((A)+(B))

int main() { puts(ADD_INT("12345",4)); return 0; }
```

[ ] Erro de pré-processamento por o 1º argumento de `ADD_INT` ser inválido.
[ ] Erro de compilação por o 1º argumento de `ADD_INT` ser inválido.
[ ] Coloca o dígito 5 no *standard output* seguido de mudança de linha.
[ ] Coloca a sequência de dígitos 52345 no *standard output* seguidos de mudança de linha.
Nota: Uma resposta errada desconta 1 valor na classificação final.

3. **[7]** Implemente a função `new_Elem` que retorna um objecto do tipo `Elem` cujo estado é iniciado com base nos argumentos de entrada passados à função. O parâmetro `i` contém o identificador do novo `Elem` constituído sempre por 10 caracteres, o parâmetro `l` define o número de inteiros que, a par de `v`, deverão ser guardados no *array* `values`. Admita que `v` aponta para um *array* cujo espaço possa ter sido alocado automaticamente. O objecto retornado deverá apresentar no campo `sumtotal` o resultado da soma dos inteiros presentes em `values`. Para a implementação de `new_Elem` considere ainda que a função `free_Elem` liberta todo o espaço ocupado por `Elem` incluindo os inteiros presentes em `values`.

```
typedef struct elem {
      char id[10];
      int sumtotal;
      int len;
      int values[0];
} Elem;

void free_Elem(Elem * e) {
      free(e);
}
```

```
Elem * new_Elem(char i[10], int l, int *v) {




}
```

3.b **[3]** Indique o resultado da expressão `sizeof(Elem)` = _____

Bom trabalho!