Instituto Superior de Engenharia de Lisboa

Licenciatura em Engenharia Informática e de Computadores

Programação em Sistemas Computacionais

Teste Global de 1ª Época, Verão de 2012/2013

---

1. [2] Escreva a função count_words, que retorna quantas palavras existem na *string* str, considerando-se como palavra qualquer sequência com um ou mais caracteres que não contenha sep.

```
unsigned count_words(const char * str, char sep);
```

2. [2,5] Considerando as definições seguintes, apresente uma versão de getInfoValue em *assembly* IA-32.
NOTA: Embora não seja obrigatório, os docentes agradecem que procure reduzir o número de instruções.

```
typedef struct info_item {
   int id;
   char stamp[12];
   unsigned info_len; /* comprimento de info, em bytes */
   char * info;
   float value;
} InfoItem;

typedef struct info_coll { unsigned num_items; InfoItem items[32]; } InfoCollection;

int getInfoValue(InfoCollection * col, unsigned idx, float * res) {
   if (idx >= col->num_items) return -1;
   *res = col->items[idx].value;
   return col->items[idx].id;
}
```

3. [4] Desenvolva, em *assembly* IA-32, a função convertAll, que aplica a função referida por convertOne a cada elemento do *array* input, para que esta deixe o resultado da conversão no índice equivalente do *array* output. Ambos os *arrays* têm espaço para n elementos. No *array* input cada elemento tem idim *bytes*, enquanto no *array* output cada elemento tem odim *bytes*. O ciclo de conversões pára se a função convertOne retornar zero, que indica que a conversão falhou. A função convertAll retorna o número de conversões realizadas com sucesso. A função referida por convertOne converte o elemento apontado pelo seu segundo argumento (src) para o espaço apontado pelo seu primeiro argumento (dst).

```
size_t convertAll(void * output, void * input, size_t n, size_t odim, size_t idim,
                  int (*convertOne)(void * dst, void * src));
```

4. [1,5] Numa cache *n-way set-associative* de 4 vias e 2^{13} (=8192) *sets*, cada linha tem exactamente a mesma dimensão do que uma instância de Entry.
NOTA: admita que os tipos têm as mesmas dimensões do que no ambiente de referência da unidade curricular.

```
typedef struct entry { int mark; char code[8]; unsigned num; float samples[12];  } Entry;
```

   a. [1] Indique, apresentando os cálculos devidamente comentados, a dimensão total da *cache*.

   b. [0,5] Que condição adicional é necessária para que, num *array* de Entry, cada índice corresponda exactamente a uma (e só uma) linha da *cache*.

5. [2] Considere os ficheiros *f1.c* e *f2.c* que, após compilação e ligação, deveriam constituir um programa (ainda que de utilidade discutível).
   a. [1] Apresente as listas de símbolos presentes em *f1.o* e em *f2.o*, que resultam da compilação de *f1.c* e *f2.c*, respectivamente. Indique as secções para os símbolos definidos.
   b. [1] Explique em detalhe como determinar o valor retornado por main.
   NOTA: não é necessário calcular o valor

f1.c

```
static int a = 3;
char b[4] = "abc";

int func(int b, char * s,
         int a, int z);

int main() {
   int c = a + 7;
   return func(a,b,c,73);
}
```

f2.c

```
int a = 21;
extern int b;

int func(int c, int x, int z) {
   return a+b+c+z;
}
```

6. [1] Um programa em execução carrega, num dado momento, várias bibliotecas de ligação dinâmica via dlopen, de modo a que fiquem todas carregadas em simultâneo. Se todas as bibliotecas tiverem funções com o mesmo nome e o programa chamar várias vezes dlsym para um desses nomes, o valor retornado é sempre o mesmo? Pode ser o mesmo às vezes? É sempre diferente? Justifique devidamente a sua resposta.

7. [3] A função addDataNode insere um novo nó, do tipo DataNode, à cabeça da lista simplesmente ligada, não circular, cujo primeiro nó é apontado pelo ponteiro presente em *ppHead (que será NULL se a lista estiver vazia). O novo nó guarda cópias da label e da sequência de num values nos campos com os mesmos nomes. A função removeDataNode elimina o primeiro nó da lista, que é apontado por *ppHead. Implemente ambas as funções.

```
typedef struct data_node DataNode;
struct data_node { DataNode * next; char label[32]; unsigned num; float * values; };

void addDataNode(DataNode * * ppHead, char * label, unsigned num, float * values);
void removeDataNode(DataNode * * ppHead);
```

8. [4] Considere a interface StringIterator, cujas implementações iteram exactamente uma vez sobre uma sequência de *strings*. O tipo ConsoleReader permite obter uma sequência de linhas do *standard input* do programa, lidas com fgets, mas, tal como está, não pode ser usado como StringIterator. Modifique o tipo ConsoleReader para que passe a ser uma implementação de StringIterator e escreva uma função main de teste, que invoca printAll passando uma instância do tipo ConsoleReader remodelado.

```
typedef struct str_iter StringIterator;
typedef struct str_iter_methods {
   int (*moveNext)(StringIterator * this);
   char * (*current)(StringIterator * this);
} StringIteratorMethods;
struct str_iter { StringIteratorMethods * vptr; };

typedef struct console_reader { char line[256]; } ConsoleReader;

int ConsoleReader_getNextLine(ConsoleReader * reader) {
   reader->line[0] = '\0';
   while (fgets(reader->line, 256, stdin)) { if (reader->line[0] != 0) return 1; }
   return 0;
}

void printAll(StringIterator * it) { while (it->vptr->moveNext(it)) puts(it->vptr->current(it)); }
```

Duração: 2 horas e 30 minutos
ISEL, 21 de Junho de 2013