Instituto Superior de Engenharia de Lisboa
Licenciatura em Engenharia Informática e de Computadores
Programação de Sistemas Computacionais
Teste Global de Época Normal, Inverno de 2021/2022

Nas questões em que não se indiquem explicitamente outras condições, considere as características do ambiente de referência usado na unidade curricular neste semestre.

1. [2,0] Implemente em linguagem C, a função `min_size_bits` que retorna a quantidade mínima de bits necessários para representar o valor do parâmetro `value`.

   ```
   int min_size_bits( int value );
   ```

   Note que, na convenção de complemento para dois, há frequentemente uma sequência de vários bits iguais ao de maior peso, que representa o sinal; a quantidade mínima de bits é a que permite armazenar o valor, incluindo apenas um bit com a representação do sinal.

2. [2,5] Implemente em linguagem C, a função `modify_style` que modifica, na *string* indicada pelo argumento `str`, as sequências com *underline* ('_') seguido de letra minúscula. Em cada sequência destas, o *underline* é eliminado e a letra é substituída pela maiúscula correspondente. No caso de caracteres *underline* que não sejam seguidos de letra minúscula não ocorre modificação.

   ```
   void modify_style(char *str);
   ```

   Por exemplo: se a *string* original for "`name_composed_by_words`", é modificada para "`nameComposedByWords`";
   se a *string* original for "`name_and_digit_2`", é modificada para "`nameAndDigit_2`";
   se a *string* original for "`TYPICAL_MACRO_NAME`", não é modificada.

3. [4] Os tipos de dados declarados abaixo permitem definir estados de configuração de conjuntos de pinos num microcontrolador.

   ```
   struct pinctrl_pin { uint32_t pinmux; uint32_t pincfg; };
   struct pinctrl_state { struct pinctrl_pin *pins; uint8_t pin_cnt; };
   struct pinctrl_config { uint8_t state_cnt; struct pinctrl_state *states[]; };

   uint32_t get_pincfg(struct pinctrl_config *config, int state, int pin) {
       return config->states[state]->pins[pin].pincfg;
   }
   ```

   a. [2] Traduza a função `get_pincfg` para linguagem *assembly* x86-64.

   b. [2] Utilizando os tipos de dados acima, defina uma estrutura de dados de modo que a invocação de `get_pincfg(&config, 2, 2)` retorne o valor `0xaa`. (`config` é uma variável do tipo `pinctrl_config`.)

4. [4] Considere a função `list_search` e o tipo `List_node`, cujas definições em linguagem C se apresentam a seguir.

   ```
   List_node *list_search(List_node *list, const void *data,
       int (*fcmp)(const void *, const void *)) {
       for (List_node *p = list; p != NULL; p = p->next)
              if (fcmp(p->data, data) == 0)
                     return p;
       return NULL;
   }

   typedef struct list_node { struct list_node *next; void *data; } List_node;
   ```

   a. [2] Implemente a função `list_search` em *assembly* x86-64.

   b. [2] Escreva, em linguagem C, um programa de teste da função `list_search`, que procura numa lista de nomes de pessoas, um nome que comece por "Abel". No programa, deve explicitar a formação da lista com pelo menos três nomes, a definição da função de comparação, assim como o código para mostrar na consola o resultado da procura.

5. [3,0] Considere o conteúdo do ficheiro fonte `f1.c` e do ficheiro de inclusão `f2.h`, bem como a lista de símbolos do ficheiro objeto `f2.o`, obtida com a ferramenta `nm`.

```
/* f1.c */
#include <stdio.h>
#include "f2.h"

#define A_SIZE( a ) ( sizeof (a) / sizeof *(a) )

static int flag;
extern Packet sample;
Packet *last;

static Packet stuff[] = {
    {10.1, 10.1, 10.1, 5.0 },
    {10.1, 20.1, 30.1, 7.5 },
    {15.2, 15.2, 10.1, 8.2 }
};

static void func_aux( void ){
    printf("Total weight%lf\n",
        sum_weight( stuff, A_SIZE( stuff ) ) );
}

int main() {
    func_aux();
       printf( "Flag: %d\n", flag );
}
```

```
/* f2.h */
#ifndef _F2_H_
#define _F2_H_
typedef struct {
   double width, length,
          height, weight;
} Packet;
double max_width( Packet[], int );
double max_length( Packet[], int );
double max_height( Packet[], int );
double sum_weight( Packet[], int );
double max_weight( Packet[], int );
#endif
```

```
/* nm f2.o */
0000000000000000 D flag
0000000000000000 B last
0000000000000032 T max_height
0000000000000019 T max_length
0000000000000064 T max_weight
0000000000000000 T max_width
0000000000000004 C state
000000000000004b T sum_weight
```

   a. [1,5] Indique o conteúdo da tabela de símbolos do ficheiro objecto relocável `f1.o`, resultante da compilação de `f1.c`. Para cada símbolo indique o nome, a secção e o respectivo âmbito (local ou global). Propõe-se que use a notação do utilitário `nm`, em que a classificação dos símbolos com minúscula (`t`, `d`, ou `b`) indica âmbito local .

   b. [1,0] Se considerar que os ficheiros objecto `f1.o` e `f2.o` podem ser combinados pelo *linker* com sucesso para gerar um executável, apresente o subconjunto dos seus símbolos que são provenientes de `f1.o` e `f2.o`, acompanhados pela respetiva classificação como *text*, *data*, etc. (admita o caso de ligação estática). Se, pelo contrário, considerar que o *linker* falhará a combinação, diga qual a razão ou razões por que isso acontece.

   c. [0,5] Indique, justificando, se o ficheiro `f2.h` deve ser incluído no ficheiro `f2.c`, do qual se obtém `f2.o`.

6. [1] O processador Intel i3-4130 incorpora *caches* de nível 1 com 32 KiB de capacidade e organização 8 *way set associative*. Considerando que a dimensão do bloco é de 64 *byte* e que os endereços são expressos a 35 *bit*, determine, justificando, o número de *bits* usados para definir os campos *tag*, *set index* e *block byte offset*.

7. [3,5] O tipo `Queue` representa uma fila de dados genéricos (representados por `void*`). A fila é baseada numa lista duplamente ligada, não intrusiva, circular, com sentinela. A operação `queue_from_array` cria e devolve uma instância de `Queue`, cujos nós da lista interna conterão os ponteiros para dados recebidos em *array*. A operação `array_from_queue` cria e devolve um *array* cujas posições conterão os ponteiros para dados recebidos na lista. Implemente ambas as funções alocando dinamicamente a memória necessária para criar as estruturas de dados e eliminando as estruturas de dados de entrada, excepto os elementos de dados.

```
typedef struct node { struct node *next, *prev; void *data; } Node;
typedef struct queue { size_t count; Node sentinel; } Queue;

Queue *queue_from_array(void *array[], size_t size);
void **array_from_queue(Queue *queue, size_t *size_array);
```

Duração: 2 horas e 30 minutos
ISEL, 1 de Fevereiro de 2022