Instituto Superior de Engenharia de Lisboa

Licenciatura em Engenharia Informática e de Computadores

Programação de Sistemas Computacionais

Teste Global da Época Normal, Inverno de 2020/2021

(Moodle)

Pergunta 1 A

Implemente, em linguagem C, a função **match_find** que procura, na palavra **value**, o padrão de bits **pattern**, com a dimensão **size** em número de bits.

Inicia a pesquisa a partir da posição de menor peso de **value**.

Retorna a posição do bit de menor peso do padrão encontrado. Se o padrão não existir, retorna -1.

int match_find(unsigned long value, unsigned long pattern, unsigned size);

Por exemplo:

match_find(0x5555, 0xa, 4); retorna 1;

match_find(0x5757, 0xa, 4); retorna 3;

match_find(0x7777, 0xa, 4); retorna -1.

Assuma as condições de execução da plataforma de programação usada em PSC.

Pergunta 1 B

Implemente, em linguagem C, a função **match_find** que procura, na palavra **value**, o padrão de bits **pattern**, com a dimensão **size** em número de bits.

Inicia a pesquisa a partir da posição de maior peso de **value**.

Retorna a posição do bit de menor peso do padrão encontrado. Se o padrão não existir, retorna -1.

int match_find(unsigned long value, unsigned long pattern, unsigned size);

Por exemplo:

match_find(0x5555, 0xa, 4); retorna 11;

match_find(0x7575, 0xa, 4); retorna 9;

match_find(0x7777, 0xa, 4); retorna -1.

Assuma as condições de execução da plataforma de programação usada em PSC.

Pergunta 2

Implemente, em linguagem C, a função **format_name** que ajusta a formatação da *string* recebida no parâmetro **name**, contendo o nome completo de uma pessoa. Um nome completo é constituído por nomes simples, formados por sequências de caracteres alfabéticos separadas por sequências de dimensão variável, de caracteres espaço (' ') ou tabulação ('\t'). A formatação consiste em colocar a primeira letra de cada nome em maiúscula, as restantes letras em minúsculas e um único caracter espaço, como separação dos nomes. Se existirem separadores no início ou no fim devem ser eliminados.

void format_name(char *name);

Pergunta 3 A

Programe, em *assembly* x86-64, a função **select_area** cuja definição em linguagem C se apresenta a seguir. Deve integrar, como início da sua implementação, o código *assembly* apresentado.

```
typedef struct{
   int side_a, side_b;
} Rectangle;

typedef struct{
   int size;
   Rectangle **data;
} Collection;

Rectangle *select_area( Collection *col, long *area ){
   long calc = 0;
   Rectangle **arr = col->data;
   Rectangle *res = NULL;
   for( int i = col->size - 1; i >= 0; --i ){
      long curr = (long)arr[i]->side_a * (long)arr[i]->side_b;
      if( curr > calc ){
         calc = curr;
         res = arr[i];
      }
   }
   *area = calc;
   return res;
}
```

```
   .text
   .global select_area
select_area:
   xor  %rdx, %rdx
   mov  8(%rdi), %rcx
   xor  %r8, %r8
   mov  (%rdi), %rdi
   dec  %rdi
   ...
```

Perguntar 3 B

Programe, em *assembly* x86-64, a função **select_area** cuja definição em linguagem C se apresenta a seguir. Deve integrar, como início da sua implementação, o código *assembly* apresentado.

```
typedef struct{
   int base, height;
} Triangle;

typedef struct{
   Triangle **data;
   int size;
} Collection;

Triangle *select_area( Collection *col, long *area ){
   long aux = 0;
   Triangle *ptr = NULL;
   Triangle **arr = col->data;
   for( int i = col->size - 1; i >= 0; --i ){
      long tmp = (long)arr[i]->base * (long)arr[i]->height;
      if( tmp > aux ){
         aux = tmp;
         ptr = arr[i];
      }
   }
   *area = aux / 2;
   return ptr;
}
```

```
   .text
   .global select_area
select_area:
   xor  %r11, %r11
   xor  %r10, %r10
   mov  (%rdi), %r9
   mov  8(%rdi), %r8
   sub  $1, %r8
   ...
```

Pergunta 4 A

Considere a função **find** cuja definição em linguagem C se apresenta a seguir.

```
void *find(void *array, size_t array_size, size_t elem_size,
           int (*predicate)(void *, void *), void *key) {
      char *last = (char *)array + array_size * elem_size;
      for (char *p = array; p < last; p += elem_size)
            if (predicate(p, key))
                  return p;
      return NULL;
}
```

```
      .text
      .global find
find:
      push  %rbx
      push  %r15
      push  %r14
      push  %r13
      push  %r12
      movq  %r8, %rbx
      movq  %rcx, %r15
      movq  %rdx, %r14
      movq  %rdi, %r13
      ...
```

```
      .text
      .global find
find:
      mov   %rcx, %rax
      imul  %rdx
      lea   (%rdi, %rax), %r9
find_for:
      cmp   %rdi, %r9
      ...
```

a) [2,5] Programe a função **find** em *assembly* x86-64, integrando uma das variantes de código *assembly* apresentado, como o início do seu código.

b) [2,5] Escreva, em linguagem C, um programa de teste da função **find** que procura num *array* de *structs* do tipo **Person** um elemento com um nome igual ao passado como argumento ao programa de teste. No programa, deve explicitar a definição e inicialização do *array* com pelo menos três posições, a definição da função de comparação, assim como a apresentação do resultado da procura na consola.

typedef struct person { char *name, int height, int weight } Person;

Pergunta 4 B

Considere a função **find** cuja definição em linguagem C se apresenta a seguir.

```
void *find(void *array, size_t elem_size, size_t array_size,
           void *key, int (*predicate)(void *, void *)) {
      char *last = (char *)array + array_size * elem_size;
      for (char *p = array; p < last; p += elem_size)
            if (predicate(p, key))
                  return p;
      return NULL;
}
```

```
      .text
      .global find
find:
      push  %rbx
      push  %r12
      push  %r13
      push  %r14
      push  %r15
      movq  %rdi, %rbx
      movq  %rdx, %r15
      movq  %rcx, %r14
      movq  %r8, %r13
      ...
```

```
      .text
      .global find
find:
      mov   %rcx, %rax
      imul  %rdx
      lea   (%rdi, %rax), %r9
find_for:
      cmp   %rdi, %r9
      ...
```

a) [2,5] Programe a função **find** em *assembly* x86-64, integrando uma das variantes de código *assembly* apresentado como o início do seu código..

b) [2,5] Escreva, em linguagem C, um programa de teste da função **find** que procura num *array* de *structs* do tipo **Person** um elemento com um nome igual ao passado como argumento ao programa de teste. No programa, deve explicitar a definição e inicialização do *array* com pelo menos três posições, a definição da função de comparação, assim como a apresentação do resultado da procura na consola.

typedef struct person { char *name, int height, int weight } Person;

Pergunta 5 A

Considere o conteúdo dos ficheiros fonte f1.c e f2.c.

```
/* f1.c */

void substitute(int, int);

int find(int);

#define M 1000

extern int size;

static int array[M];

int f() {
      return find(M - 1);
}

int main() {
      f();
}
```

```
/* f2.c */

static int size = 2;

int f = 3, g;

int M = 20;

int array[20];

static int find(int x) {
      return array[x];
}

int substitute(int x, int y) {
      array[x] = y;
}
```

a) [1,5] Indique o conteúdo das tabelas de símbolos dos ficheiros objecto relocáveis, resultantes da compilação de f1.c e f2.c . Para cada símbolo, indique o nome, a secção e o respectivo âmbito (local ou global). (Pode usar a convenção do utilitário nm.)

b) [1,5] Identifique, justificando, quais os erros que ocorrem na ligação entre os módulos f1.o e f2.o.

Pergunta 5 B

Considere o conteúdo dos ficheiros fonte f1.c e f2.c.

```
/* f1.c */

#define N 1000

extern int size;

void assign(int, int);

int search(int);

int array[N];

int g() {
      return search(N - 1);
}

int main() {
      return g();
}
```

```
/* f2.c */

static int size = 2;

int g = 3, f;

int N = 20;

static int array[20];

static int search(int x) {
      return array[x];
}

int assign(int x, int y) {
      array[x] = y;
}
```

a) [1,5] Indique o conteúdo das tabelas de símbolos dos ficheiros objecto relocáveis, resultantes da compilação de f1.c e f2.c . Para cada símbolo, indique o nome, a secção e o respectivo âmbito (local ou global). (Pode usar a convenção do utilitário nm.)

b) [1,5] Identifique, justificando, quais os erros que ocorrem na ligação entre os módulos f1.o e f2.o.

Pergunta 6 A

```
typedef struct data {
      int class;
      size_t data_len;
      char data[MAX_DATA];
} Data;
```

```
typedef struct node {
      struct node *link;
      Data *data;
} DataNode;
```

Considere elementos de informação suportados em **struct data**, constituídos por um bloco de caracteres, definido pelos campos **data** e **data_len**, e com a classificação indicada no campo **class**.

Programe a função **data_filter** que, recebendo um *array* de elementos de informação, cria réplicas dos elementos com uma dada classificação. A função devolve as réplicas criadas na forma de uma lista ligada, baseada em elementos do tipo **DataNode**.

DataNode *data_filter(Data array[], size_t data_len, int class);

Programe a função **list_data_free** que liberta toda a memória alocada por **data_filter**.

void list_date_free(DataNode *list);

Pergunta 6 B

```
typedef struct data {
      int class;
      size_t data_len;
      char *data;
} Data;
```

```
typedef struct node {
      struct node *link;
      Data data;
} DataNode;
```

Considere elementos de informação opaca suportados na **struct data**. Um elemento de informação é constituído por um bloco de caracteres, definido pelos campos **data** e **data_len**, e tem a classificação indicada no campo **class**.

Programe a função **data_filter** que, recebendo um *array* de elementos de informação, cria réplicas dos elementos com uma dada classificação. A função devolve as réplicas criadas na forma de uma lista ligada, baseada em elementos do tipo **DataNode**.

DataNode *data_filter(Data array[], size_t data_len, int class);

Programe a função **list_data_free** que liberta toda a memória alocada por **data_filter**.

void list_date_free(DataNode *list);