Exercícios: Recursão - Recorrências - Listas

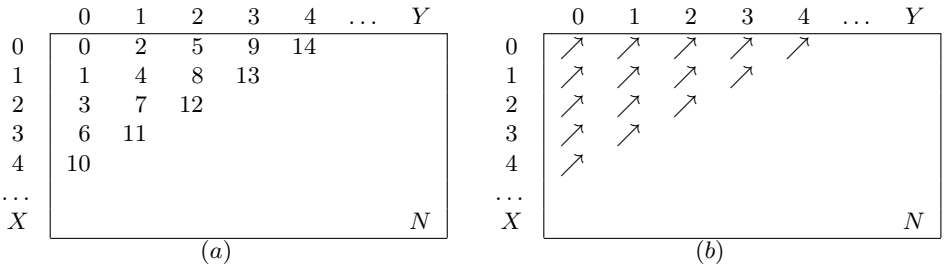
Ronaldo Fumio Hashimoto

1 Recursão

1. Simule a seguinte função recursiva para $n = 6$:

```
int zzz (int n) {
  int aux;
  if (n <= 2)
    return 1;
  n--;
  aux = zzz(n);
  n--;
  return aux + zzz (n);
}
```

2. Considere a seguinte tabela do lado esquerdo da figura abaixo:



Observe que os números na tabela foram obtidos seguindo a ordem das setas da tabela acima do lado direito.

Escreva uma função recursiva `tabela` que recebe como parâmetro um inteiro não negativo n e calcula um par de inteiros (x, y), onde x e y são as coordenadas de n na tabela abaixo:

3. Considere a seguinte função recursiva:

```
int fd (int x, int n) {
  int z;
  if (n==0) return 1;
  z = fd (x, n/2);
  z = z * z;
  if (n%2 == 1) z = z * x;
  return z;
}
```

- Simule esta função para $x = 3$ e $n = 5$.
- Se $T(n)$ denota o número de comparações `if (n==0)` feitas pela chamada da função `fd (x, n)`, escreva a fórmula de recorrência para $T(n)$.
- Prove, por indução em n, que $T(n) = 1 + \log_2 n$.

2 Listas ligadas

Observação: considere que todas as listas ligadas dos exercícios desta seção são implementadas usando vetores de células estáticos como mostrada na Aula 12:

```
typedef struct celula {
  int conteudo;
  int prox;
} Celula;

# define MAX 100

Celula v[MAX];
int inicio, livre;
```

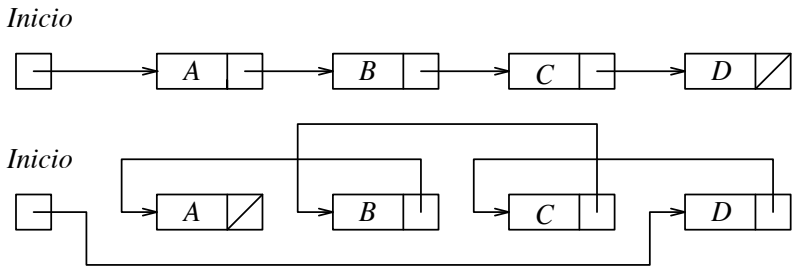
1. Escreva uma função que recebe uma lista ligada com números inteiros ordenados crescentemente, e remova da lista os elementos repetidos, deixando apenas uma cópia de cada elemento.

2. Escreva funções para as seguintes operações:

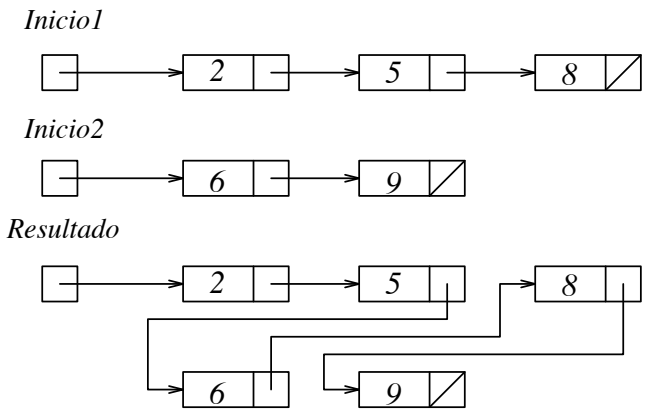
   - verifica se um dado x ocorre em uma lista ligada;
   - acrescenta um elemento x no fim de uma lista ligada;
   - remove o último elemento de uma lista ligada.

3. Faça uma função recursiva que devolve o índice do elemento do meio de uma lista ligada (se o número de elementos da lista é par, devolve o índice do $\frac{n}{2}$-ésimo elemento da lista - se é ímpar, também) **sem contar o número de elementos da lista**.

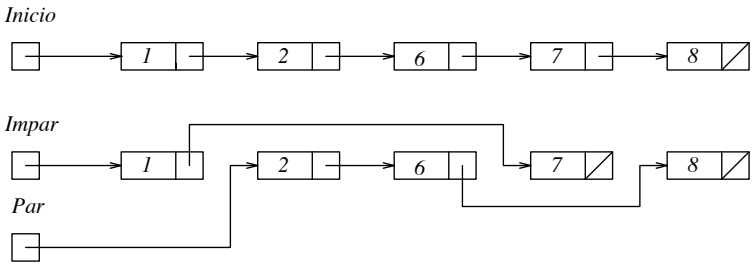
4. Escreva uma função recursiva que inverte uma lista ligada dada (o primeiro elemento da nova lista é o último da lista dada, o segundo é o penúltimo da lista dada, e assim por diante. Faça manipulando apenas os campos `prox`. Exemplo:



5. Escreva uma função para intercalar duas listas ligadas cujas informações estão arranjadas em ordem crescente. Exemplo:



6. Dada uma lista ligada escreva uma função que transforma a lista dada em duas listas ligadas: a primeira contendo os elementos cujos conteúdos são pares e a segunda com os elementos cujos conteúdos são ímpares. Sua função deve manipular somente os campos `prox` e **não** o conteúdo das células (isto é, não vale ficar copiando os conteúdo de um lado para o outro, só vale alterar os valores do campo `prox`). Exemplo:



3 Exercício prático

1. Estude atentamente o programa abaixo:

```
# include <stdio.h>
# include <string.h>

int read_word (FILE * arq, char * word) {
  /* This function reads a sequence of characters that contains only alphabetic */
  /* letters and underscore */
  /* For this exercise, you really do not need to understand fully this function */
  /* Just use it */
  fscanf (arq, "%[^a-zA-Z_]", word);
  return fscanf (arq, "%[a-zA-Z_]", word);
}

int reserved (char *word, char ** reserved_words) {
  /* This function returns 1 if word is a reserved word; 0, otherwise */
  if (strcmp ("", *reserved_words) == 0) return 0;
  if (strcmp (word, *reserved_words) == 0) return 1;
  return reserved (word, reserved_words+1);
}

/* This program reads a C program and count the number of reserved words */
int main (int argc, char * argv[]) {
  FILE * arq;
  char * reservadas [] = {
    "auto", "break",  "case", "char",  "const", "continue", "default", "do",
    "double", "else", "enum", "extern", "float", "for", "goto", "if", "int",
    "long", "register", "return", "short", "signed", "sizeof", "static", "struct",
    "switch", "typedef", "union", "unsigned", "void", "volatile", "while", ""};
  int count;
  char word[1000], *filename;
  if (argc != 2) {
    printf ("Use: %s <source C programa filename>\n", argv[0]);
    return 0;
  }
  filename = argv[1];
  arq = fopen (filename, "r");
  if (arq == NULL) {
    printf ("File %s could not be found\n", filename);
    return 0;
  }
  count = 0;
```

```
  while (read_word (arq, word) == 1)
    if (reserved (word, reservadas) == 1) count++;
  printf ("File %s contains %d reserved words\n", filename, count);
  fclose (arq);
  return 0;
}
```

Se você ainda estiver em dúvida, salve este programa em um arquivo, por exemplo de nome `conta.c`, compile e rode da seguinte forma:

```
gcc conta.c -o conta
./conta conta.c
```

O resultado será a impressão:

```
File conta.c contains 58 reserved words
```

Isto indica que o programa `conta.c` contém 58 palavras reservadas.

2. Utilizando a estrutura:

```
typedef struct celula {
  char word [80];
  int count;
  int prox;
} Celula;
```

E a lista ligada implementada como vetor estático:

```
# define MAX 100
Celula v[MAX];
int inicio, livre;
```

Modifique o programa acima para contar a frequência de cada palavra reservada e não-reservada. A sua saída deverá ser, por exemplo:

```
File conta.c contains the following reserved words:
        auto apareceu 1 veze(s)
        break apareceu 1 veze(s)
        case apareceu 1 veze(s)
        char apareceu 7 veze(s)
        const apareceu 1 veze(s)
        continue apareceu 1 veze(s)
        default apareceu 1 veze(s)
        do apareceu 2 veze(s)
        double apareceu 1 veze(s)
        else apareceu 1 veze(s)
        enum apareceu 1 veze(s)
        extern apareceu 1 veze(s)
        float apareceu 1 veze(s)
        for apareceu 1 veze(s)
        goto apareceu 1 veze(s)
        if apareceu 7 veze(s)
        int apareceu 6 veze(s)
        long apareceu 1 veze(s)
        register apareceu 1 veze(s)
        return apareceu 8 veze(s)
        short apareceu 1 veze(s)
        signed apareceu 1 veze(s)
        sizeof apareceu 1 veze(s)
        static apareceu 1 veze(s)
        struct apareceu 1 veze(s)
```

```
        switch apareceu 1 veze(s)
        typedef apareceu 1 veze(s)
        union apareceu 1 veze(s)
        unsigned apareceu 1 veze(s)
        void apareceu 1 veze(s)
        volatile apareceu 1 veze(s)
        while apareceu 2 veze(s)
Total: 58 reserved words.

File conta.c contains the following non-reserved words:
        include apareceu 2 veze(s)
        stdio apareceu 1 veze(s)
        h apareceu 2 veze(s)
        string apareceu 1 veze(s)
        read_word apareceu 2 veze(s)
        FILE apareceu 2 veze(s)
        arq apareceu 8 veze(s)
        word apareceu 11 veze(s)
        This apareceu 3 veze(s)
        function apareceu 3 veze(s)
        reads apareceu 2 veze(s)
        a apareceu 5 veze(s)
        sequence apareceu 1 veze(s)
        of apareceu 2 veze(s)
        characters apareceu 1 veze(s)
        that apareceu 1 veze(s)
        contains apareceu 2 veze(s)
        only apareceu 1 veze(s)
        alphabetic apareceu 1 veze(s)
        letters apareceu 1 veze(s)
        and apareceu 2 veze(s)
        underscore apareceu 1 veze(s)
        For apareceu 1 veze(s)
        this apareceu 2 veze(s)
        exercise apareceu 1 veze(s)
        you apareceu 1 veze(s)
        really apareceu 1 veze(s)
        not apareceu 2 veze(s)
        need apareceu 1 veze(s)
        to apareceu 1 veze(s)
        understand apareceu 1 veze(s)
        fully apareceu 1 veze(s)
        Just apareceu 1 veze(s)
        use apareceu 1 veze(s)
        it apareceu 1 veze(s)
        fscanf apareceu 2 veze(s)
        zA apareceu 2 veze(s)
        Z_ apareceu 2 veze(s)
        reserved apareceu 6 veze(s)
        reserved_words apareceu 4 veze(s)
        returns apareceu 1 veze(s)
        is apareceu 1 veze(s)
        otherwise apareceu 1 veze(s)
        strcmp apareceu 2 veze(s)
        program apareceu 2 veze(s)
        C apareceu 2 veze(s)
        count apareceu 5 veze(s)
        the apareceu 1 veze(s)
        number apareceu 1 veze(s)
        words apareceu 2 veze(s)
        main apareceu 1 veze(s)
```

```
        argc apareceu 2 veze(s)
        argv apareceu 3 veze(s)
        reservadas apareceu 2 veze(s)
        filename apareceu 6 veze(s)
        printf apareceu 3 veze(s)
        Use apareceu 1 veze(s)
        s apareceu 3 veze(s)
        source apareceu 1 veze(s)
        programa apareceu 1 veze(s)
        n apareceu 3 veze(s)
        fopen apareceu 1 veze(s)
        r apareceu 1 veze(s)
        NULL apareceu 1 veze(s)
        File apareceu 2 veze(s)
        could apareceu 1 veze(s)
        be apareceu 1 veze(s)
        found apareceu 1 veze(s)
        d apareceu 1 veze(s)
        fclose apareceu 1 veze(s)
Total: 140 non-reserved words.
```

Para exercício vai exigir que você manipule *strings*. Para isso, estude com cuidado o material que se encontra em

`http://www.ime.usp.br/~pf/algoritmos/aulas/string.html`

Especialmente a Seção "A biblioteca string" para o uso das funções *strcmp* e *strcpy*.