

Paradigmas de Programação

# **Programação Funcional**

Prof<sup>a</sup> Andréa Schwertner Charão

DLSC/CT/UFSM

# Sua solução para a empresa QueroXis

*“Programa para gerar a lista de itens em HTML. A entrada do seu programa é a lista de TODOS os nomes das clientes, enviada pela empresa. A saída é a lista em HTML, em ordem alfabética, só com os nomes que iniciam com "R".”*

# Sua solução para a empresa QueroXis

*“Programa para gerar a lista de itens em **HTML**.*

strings <ul><li>,  
<table><tr><td>, ..

*A **entrada** do seu*

fopen, fscanf,  
fgets, ..

*programa é a **lista** de*

char\*  
struct / array [ ]

*TODOS os nomes das clientes, enviada pela empresa. A **saída** é a*

fputs, fprintf,  
fclose, ..

*lista em HTML, em **ordem** alfabética, só*

qsort,  
função de  
comparação

*com os nomes que **iniciam com "R"**.*”

if (name[0] == 'R')

# Muitas soluções em C



- Ações (programação **procedimental, imperativa**)
  - ler entrada
  - selecionar
  - ordenar
  - escrever saída
- Quantidade de linhas de código (.c, .h)
  - 232, 65, 122, 37, ..
- Quantidade de procedimentos
  - 8, 2, 5, 2, ..

# Trechos de solução em C

```
int main(){
    int QDN/*Quantidade de Nomes*/,TAM=200;
    printf("Insira a quantidade de nomes\n");
        scanf("%d",&QDN);
    char aux[TAM]/*Variavel auxiliar para ler os nomes do arquivo e salvar no vetor de string*/,NameList[QDN][TAM];
    FILE *entrada;

        entrada = fopen("ListaNomes.txt", "r");
        for(int i=0;i<QDN;i++){
fgets(aux,TAM,entrada);
strcpy (NameList[i],aux);        /* Copia str1 em str2 */
                                }

        qsort(NameList,QDN,TAM/*bytes*/, compara);
}
```

- (I)Legibilidade (indent) 
- Arrays de char, tamanhos fixos
- Ordenação antes de seleção 

Mais fácil 

Mais limitado 

# Trechos de solução em C



```
#define DEFAULT_VECTOR_SIZE 32 // The sta
#define MAX_NAME_LEN 1024 // The line max

struct names_list_s
{
    /* Using an dynamically allocated
    char **array;
    int size;
};

FILE *fd = fopen(path, "rt");
if (fd == NULL)
{
    return NULL;
}

char *str;
names_list_t *names = (names_list_t*)malloc(sizeof(names_list_t));
names->array = (char**)malloc(sizeof(char*)*DEFAULT_VECTOR_SIZE);

names->size = 0;
while (!feof(fd))
{
    str = get_line(fd);
    if (str == NULL)
        continue;
    if ((str[0] == 'R') || (str[0] == 'r'))
```

- Código organizado 
- Arrays de char, aloc. dinâmica
- Seleção antes de ordenação 

Mais  
flexibilidade  
Mais código



# Trechos de solução em C



```
for (i=0; i<names->size; i++)
{
    if (!write_string(fd, "<tr>\n\t<td align=center>"))
    {
        goto bad_return;
    }
    if (!write_string(fd, names->array[i]))
    {
        goto bad_return;
    }
    if (!write_string(fd, "\t</td>\n</tr>\n"))
    {
        goto bad_return;
    }
}
if (!write_string(fd, "</table>\n</body>\n</html>\n\n"))
{
    goto bad_return;
}

fclose(fd);
return true;

bad_return:
fclose(fd);
return false;
```

# Trechos de solução em C

```
for (i=0; i<names->size; i++)
{
    if (!write_string(fd, "<tr>\n\t<td align=center>"))
    {
        goto bad_return;
    }
    if (!write_string(fd, names->array[i]))
    {
        goto bad_return;
    }
}
```

- *Goto is considered harmful* 
- Aqui, usado para tratar exceção 
- Mesmo assim, é muito código só para tratar I/O





# Programação funcional

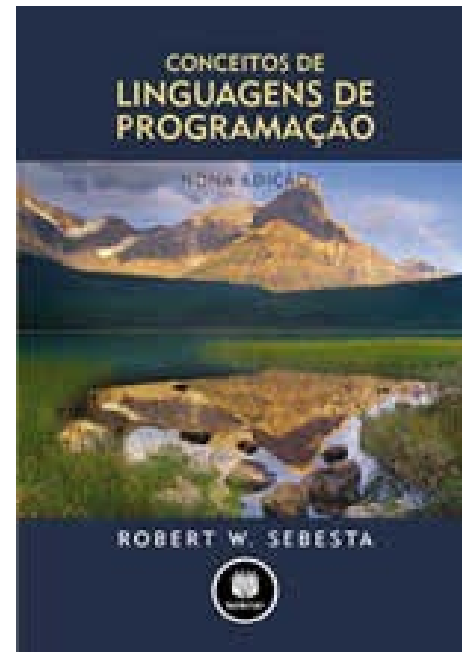
- [E-book] Sebesta, R.

## **Conceitos de Linguagens de Programação.**

Bookman, 2011.

Capítulo 15:

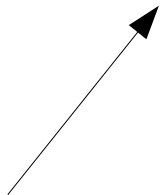
Linguagens de  
programação  
funcional.



# Solução funcional

- Filtrar a lista com condição “head str == 'R'”
- Ordenar
- Aplicar tags a todos nomes selecionados

```
let lista = ["Maria", "Renata", "Fabiana", "Rosangela"]  
  
in sort (filter (\n -> head n == 'R') lista)
```



Filtrar a lista com condição

# Solução funcional

## ■ Hey, e as tags HTML?

```
let lista = ["Maria", "Renata", "Fabiana", "Rosangela"]  
  
in map (\s -> "<LI>" ++ s ++ "</LI>") (sort (...))
```

Aplica função para todos elementos da lista



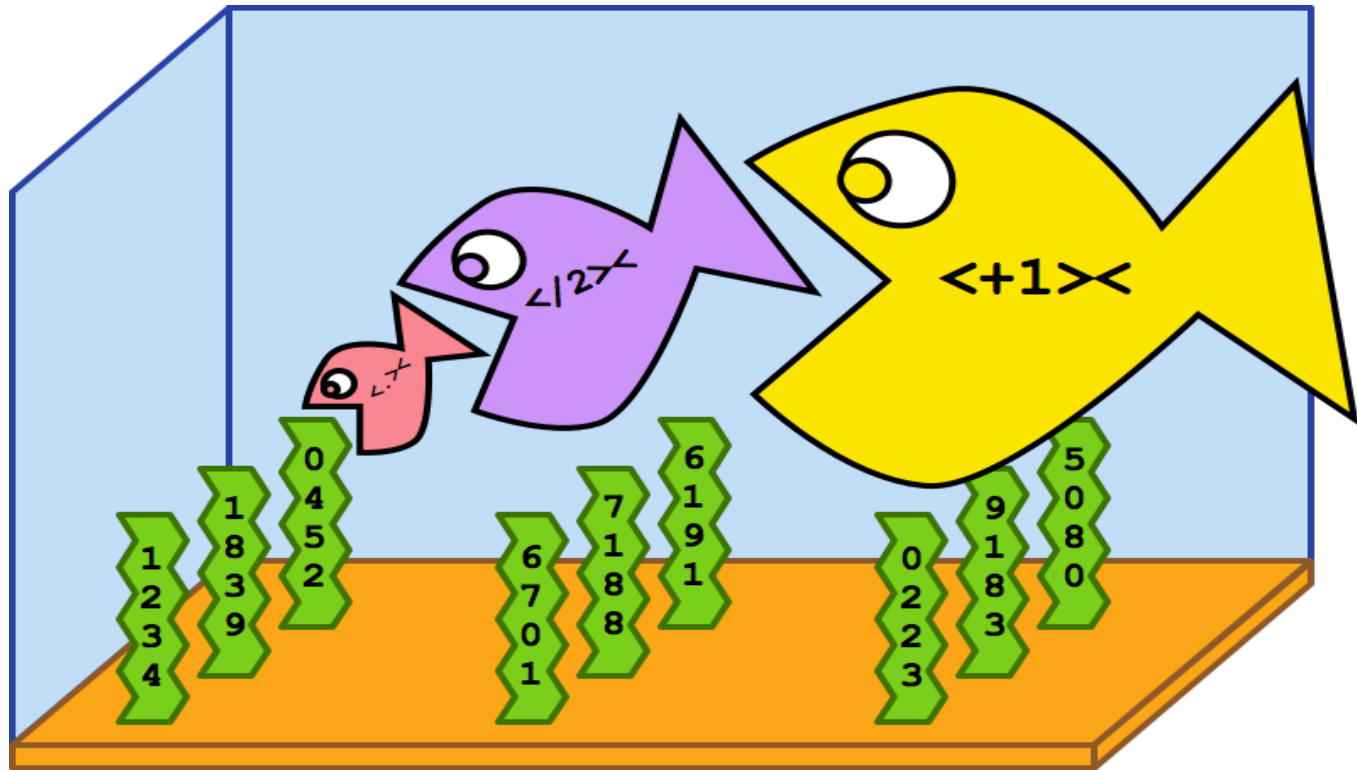
Concatenação de strings



# Solução funcional

- Hey, e as operações de I/O ???
- Separação:
  - lógica X entrada/saída  
(interação/apresentação)
- Mais 5 linhas de código

# Solução funcional



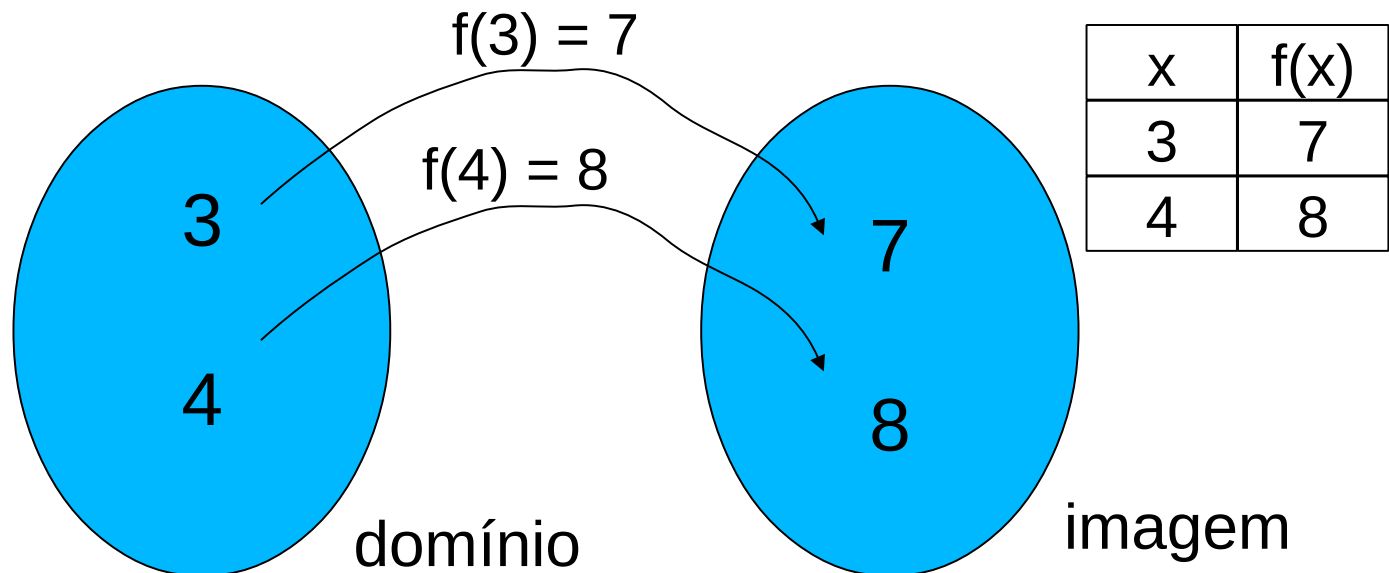
[http://www.huffingtonpost.com/john-pavley/teach-a-kid-functional-pr\\_b\\_3666853.html](http://www.huffingtonpost.com/john-pavley/teach-a-kid-functional-pr_b_3666853.html)

# Programação funcional

- Paradigma inspirado em funções matemáticas, p.ex.:

$$f(x) = x + 4$$

- Função: é uma correspondência entre membros de 2 conjuntos



# Operações com funções

- Definição

$$\text{Ex.: } f(x) = x + 4$$

- Aplicação

$$\text{Ex.: } f(3)$$

- Composição

$$\text{Ex.: } f \circ g(x) = f(g(x))$$

- Função condicional

$$\text{Ex.: } f(x) = \begin{cases} 1 & \text{se } x = 2 \\ x + 1 & \text{se } x > 2 \end{cases}$$

# Operações com funções

- Definição recursiva

$$\text{Ex.: } f(x) = \begin{cases} 1 & \text{se } x = 0 \\ x * f(x-1) & \text{se } x > 0 \end{cases}$$

Ex.:

$$0! = 1$$

$$n! = n * (n-1)$$



# Mais sobre programação funcional

- Sem variáveis
- Funções não têm "efeitos colaterais"
- Funções de alta ordem (higher-order)
- Programação fortemente modular
- Fluxo de controle não detalhado pelo programador

# Linguagens de programação funcional

## Mother Tongues

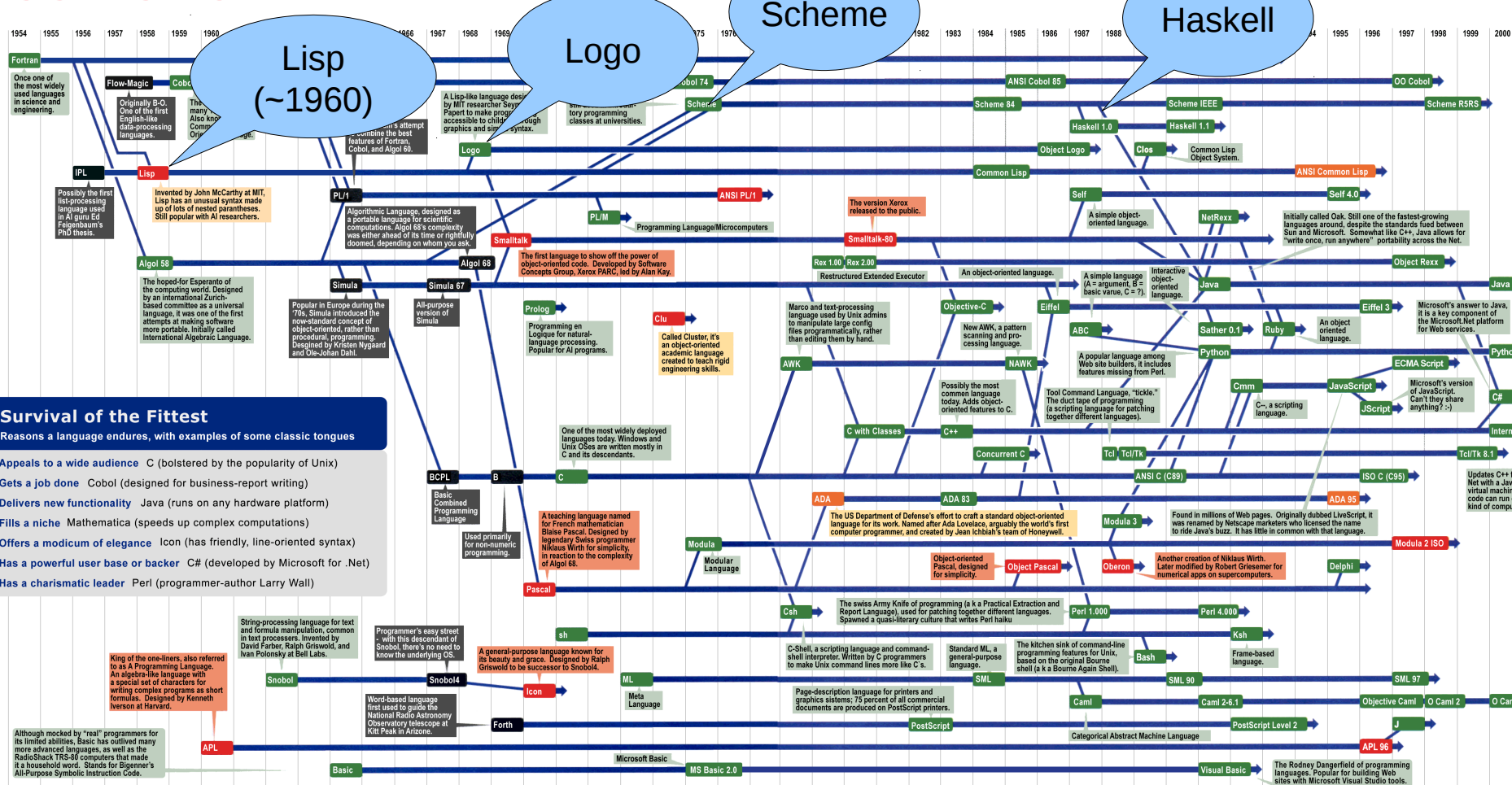
Tracing the roots of computer languages through the ages

Just like half of the world's spoken tongues, most of the 2,300-plus computer programming languages are either endangered or extinct. As powerhouses C/C++, Visual Basic, Cobol, Java and other modern source codes dominate our systems, hundreds of older languages are running out of life.

An ad hoc collection of engineers-electronic lexicographers, if you will-aim to save, or at least document the lingo of classic software. They're combing the globe's 9 million developers in search of coders still fluent in these nearly forgotten lingua frangas. Among the most endangered are Ada, APL, B (the predecessor of C), Lisp, Oberon, Smalltalk, and Simula.

Code-raker Grady Booch, Rational Software's chief scientist, is working with the Computer History Museum in Silicon Valley to record and, in some cases, maintain languages by writing new compilers so our ever-changing hardware can grok the code. Why bother? "They tell us about the state of software practice, the minds of their inventors, and the technical, social, and economic forces that shaped history at the time," Booch explains. "They'll provide the raw material for software archaeologists, historians, and developers to learn what worked, what was brilliant, and what was an utter failure." Here's a peek at the strongest branches of the family tree. For a nearly exhaustive rundown, check out [http://www.fh-frankfurt.de/Java/misc/lang\\_hist.html](http://www.fh-frankfurt.de/Java/misc/lang_hist.html).

1954	Year Introduced
Active: thousands of users	
Protected: taught at universities; compilers available	
Endangered: usage dropping off	
Extinct: no known active users or up-to-date compilers	
Lineage continues	



### Survival of the Fittest

Reasons a language endures, with examples of some classic tongues

Appeals to a wide audience C (bolstered by the popularity of Unix)

Gets a job done Cobol (designed for business-report writing)

Delivers new functionality Java (runs on any hardware platform)

Fills a niche Mathematica (speeds up complex computations)

Offers a modicum of elegance Icon (has friendly, line-oriented syntax)

Has a powerful user base or backer C# (developed by Microsoft for .Net)

Has a charismatic leader Perl (programmer-author Larry Wall)

# Origens da programação funcional

- LISP (LISt Processor)
  - MIT (1960)
  - processamento de listas
  - manipulação de símbolos
  - aplicações em inteligência artificial
- Vários dialetos da linguagem
  - CommonLisp, Scheme, Goldenlisp, etc.
- Utilizada em programas 'avançados'
  - Ex.: editor Emacs, scripting no GIMP, etc.

# Outras linguagens

- Erlang
- F#
- Javascript
- Python
- Ruby
- Scala
- Etc.

Possuem suporte a algumas características da programação funcional, mas não são puramente funcionais.

Paradigmas de Programação

**Nossa escolha  
para este semestre!**

# **Linguagem Haskell**

Profª Andréa Schwertner Charão  
DLSC/CT/UFSM