Programas

Teoria da Computação

INF05501

Programas, Máquinas e Computação

- Em geral
  - **Programas** são descritos usando uma **linguagem de programação**
  - **Máquinas** são descritas através de uma **arquitetura**
  - **Computações** são definidas como a **relação** entre o programa e a máquina

- **Inviável formalizarmos os conceitos** de programas e máquinas **com alguma linguagem específica e um computador real**

- Para isto, utilizamos **modelos matemáticos simples**, os quais incluem as **características essenciais** de programas e máquinas

Formalização de Programas

- Um programa é um conjunto de **instruções**, que capacitam uma máquina a transformar **dados iniciais** em um **resultado desejado**
- Estas instruções são compostas de acordo com uma **estrutura de controle**
- Para estudarmos programas, **não** precisamos saber qual **operação concreta** é representada por uma instrução
- Portanto, podemos usar **identificadores** a fim de diferenciá-las

Instruções

- Instruções podem definir operações básicas ou testes
  - **Operações básicas** podem ter **resultados variados**
  - **Testes** só podem ter um **resultado booleano** (verdadeiro - V ou falso - F)
- Por isto, definiremos
  - Um conjunto de **identificadores de operações** (Ex.: $P, Q, R, ...$), e
  - Um conjunto de **identificadores de testes** (Ex.: $T_1, T_2, T_3, ...$)
- Existe ainda uma operação especial ✓ chamada de operação vazia, a qual **não possui efeito algum**

Composição de Instruções

- Independentemente do tipo de estruturação, instruções podem ser **compostas** das seguintes maneiras:
  - Composição Sequencial: cada instrução é executada seguindo-se uma **ordenação**, sendo que a instrução seguinte só é executada após o término da atual
  - Composição Não-Determinística: a partir de um conjunto de instruções compostas, uma delas é **aleatoriamente selecionada** para executar
  - Composição Concorrente: todas as instruções compostas podem ser executadas em **qualquer ordem, inclusive simultaneamente**

Tipos de Programas

- O tipo da estrutura de controle define o **tipo de programa** segundo a seguinte classificação:
  - Monolítico: baseado em **desvios** condicionais e incondicionais
  - Iterativo: baseado em **estruturas de iteração** de partes do programa
  - Recursivo: baseado em **sub-rotinas recursivas**
- Veremos cada tipo em mais detalhes a seguir

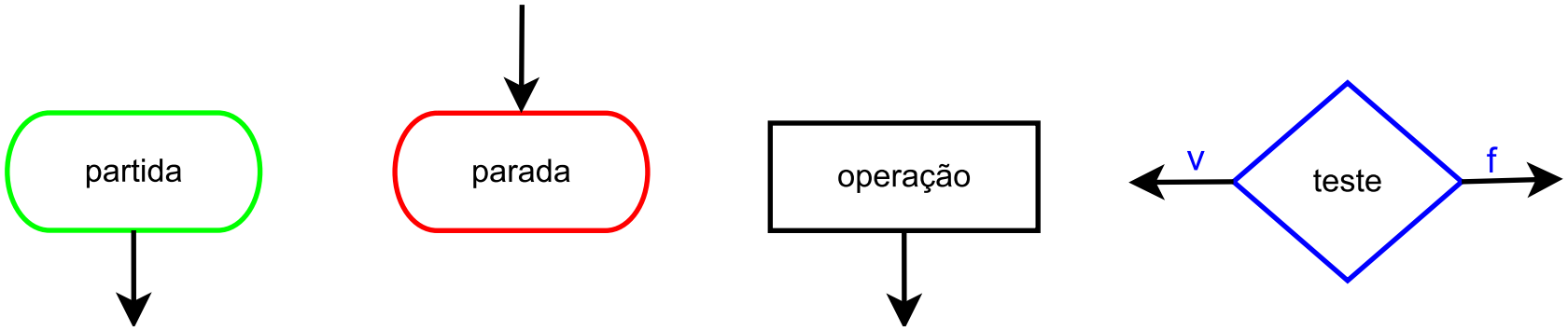
Programas Monolíticos

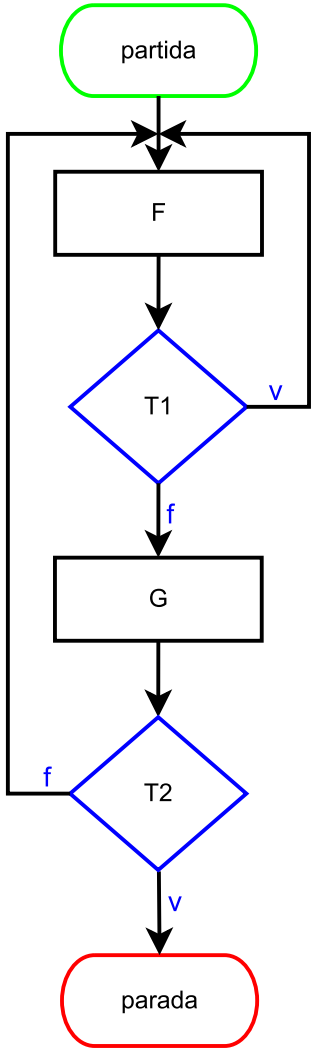
- Um programa monolítico é estruturado usando **desvios condicionais e incondicionais**
- **Não** faz uso explícito de **mecanismos auxiliares de programação**, tais como iteração, sub-rotinas e recursão
- Desta forma, a **lógica** é **distribuída** por todo o bloco (monolito) que constitui o programa

Representação por Fluxogramas

- A forma mais convencional de se representar um programa monolítico é através de fluxogramas

- Fluxogramas são diagramas na forma de **grafos direcionados** em que
  - Os **nodos** correspondem a **instruções**
  - Os **arcos** definem **sequências de instruções**




partida
F
T1
v
f
G
T2
f
v
parada

Representação por Instruções Rotuladas

- Outra forma de representar programas monolíticos é através de instruções rotuladas
- Elas são uma **descrição textual** do quê é representado em **fluxogramas**
- Como indica o nome, cada instrução possui um rótulo, o qual serve como um **identificador unívoco** de forma que possamos referenciar instruções dentro de um programa

Representação por Instruções Rotuladas (cont.)

- Cada tipo de instrução rotulada representa um elemento de um fluxograma:
  - **Operação:** Indica a operação a ser executada, seguida de um desvio incondicional para a instrução seguinte
  - **Teste:** Determina um desvio condicional, determinado pela avaliação de um teste
  - **Parada:** Definida por um desvio incondicional para um rótulo que não possui uma instrução correspondente
  - **Partida:** Assume-se que a computação sempre se inicia no rótulo 1

Representação por Instruções Rotuladas (cont.)

- **Exemplo:**

```
1: faça F vá_para 2
2: se T1 então vá_para 1 senão vá_para 3
3: faça G vá_para 4
4: se T2 então vá_para 5 senão vá_para 1
```

Definição Formal de Instruções Rotuladas

- Um rótulo (ou etiqueta) é uma **cadeia finita de caracteres**, composta por letras e números

- Uma instrução rotulada i é uma **sequência de símbolos** de uma das duas seguintes formas, onde F é um identificador de operação, T, um identificador de teste e r_1, r_2 e r_3 são rótulos:

  - Operação: r_1 : `faça` F `vá_para` r_2
  - Teste: r_1 : `se` T `então vá_para` r_2 `senão vá_para` r_3

Definição Formal de Programa Monolítico

- Usando instruções rotuladas, fica mais fácil definirmos um programa monolítico de maneira formal:

  *Um **programa monolítico** é um par ordenado $P = (I, r_0)$, onde*

  – I *é um conjunto finito de instruções rotuladas*

  – r_0 *é chamado de rótulo inicial, o qual está associado à primeira instrução do programa*

Definição Formal de Programa Monolítico (cont.)

- **Não** existem **duas instruções diferentes** associadas a um **mesmo rótulo**
- Se um rótulo **é referenciado** em uma instrução **mas não possui uma instrução associada**, então ele é dito um rótulo final
- Note: a definição de programa monolítico requer que exista pelo menos uma instrução, a qual é associada ao rótulo inicial

Exercício

1. Apresente o fluxograma correspondente ao programa monolítico abaixo, descrito usando instruções rotuladas:

```
1: faça O1 vá_para 2
2: se T1 então vá_para 3 senão vá_para 4
3: faça O2 vá_para 4
4: se T2 então vá_para 5 senão vá_para 1
```

Programas Iterativos

- Um programa iterativo é descrito usando-se **estruturas de iteração** de partes do programa
- O uso de estruturas de controle iterativas é uma tentativa de **melhorar o entendimento** e **facilitar a manutenção** de programas
- Desvios incondicionais são substituídos por estruturas de ciclos e repetições $\Rightarrow$ programação estruturada
- Utiliza-se de três **mecanismos de composição** (sequencial) de programas encontrados em muitas linguagens de programação de alto nível

Mecanismos de Composição Sequencial

- Sequencial: Composição de programas P e P', resultando em um programa P'', o qual representa a **execução de P seguida da execução de P'**

- Condicional: Composição de programas P e P' que resulta em um programa P'', o qual representa a **execução ou de P ou de P'**, dependendo do resultado de um teste

- Repetição: Composição de um programa P que resulta em um programa P', o qual representa a **repetição da execução** de P a partir do resultado de um teste

Mecanismos de Composição Sequencial (cont.)

- A **composição de repetição** possui duas formas:
  - Enquanto: **Repetição** da execução de P **enquanto** o teste for **verdadeiro**
  - Até: Análoga à anterior, mas a **repetição** ocorre **enquanto** o resultado do teste for **falso**

Definição Formal de Programa Iterativo

*Um **programa iterativo** P é indutivamente definido da seguinte forma:*

- *A operação vazia ✓ é um programa iterativo*
- *Todo identificador de operação é um programa iterativo*
- *Sendo F e G programas iterativos e T um identificador de teste, então também são programas iterativos:*
  - F;G
  - `(se` T `então` F `senão` G`)`
  - `enquanto` T `faça (`F`)`
  - `até` T `faça (`F`)`

Definição Formal de Programa Iterativo (cont.)

- **Parênteses** são usados para **retirar as ambiguidades de interpretação**
- Note que o mecanismo de repetição **pode ter duas interpretações**:
  - (enquanto T faça F);G
  - enquanto T faça (F;G)

Exemplo de Programa Iterativo

```
(se T1
então enquanto T2
      faça (até T3
            faça (F;G))
senão (✓))
```

Qual é o fluxograma equivalente a este programa iterativo?

Exemplo de Programa Iterativo (cont.)

- Note que a **tradução** de **programa iterativo para fluxograma** é **trivial**
- No entanto, veremos mais adiante, que encontrar um **programa iterativo equivalente a um fluxograma nem sempre é possível para qualquer máquina**
- **Programas iterativos são facilmente traduzidos em programas monolíticos equivalentes**
- Como poderíamos traduzir o programa iterativo do exemplo em um programa monolítico equivalente?

Programas Recursivos

- Um programa recursivo é aquele que inclui a utilização de **sub-rotinas recursivas**
- Recursão é uma forma **indutiva** de definir programas
- Sub-rotinas permitem a **estruturação hierárquica** de programas, possibilitando **níveis diferenciados de abstração**
- Para usarmos sub-rotinas, definimos um conjunto de identificadores de sub-rotinas $\{\mathcal{R}_1, \mathcal{R}_2, ...\}$

Definição Formal de Expressão de Sub-Rotina

*Uma **expressão de sub-rotina (ou somente expressão)** E é definida indutivamente da seguinte maneira:*

- *A operação vazia ✓ é uma expressão de sub-rotina*
- *Todo identificador de operação é uma expressão de sub-rotina*
- *Todo identificador de sub-rotina é uma expressão de sub-rotina*
- *Sendo E_1 e E_2 expressões de sub-rotinas e T um identificador de teste, então também são expressões de sub-rotinas:*
  - $E_1;E_2$
  - `(se` T `então` E_1 `senão` E_2`)`

Definição Formal de Programa Recursivo

*Um **programa recursivo** P possui a seguinte forma:*

$$P \texttt{ é } E_0 \texttt{ onde } \mathcal{R}_1 \texttt{ def } E_1, \mathcal{R}_2 \texttt{ def } E_2, \ \ldots, \ \mathcal{R}_n \texttt{ def } E_n$$

onde, para $k \in \mathbb{N}$ e $1 \leq k \leq n$:

- E_0 *é a expressão inicial, a qual é uma expressão de sub-rotina*
- E_k *é a expressão que define* $\mathcal{R}_k$
- *A operação vazia* ✓ *é um programa recursivo que não faz coisa alguma*

Exemplo de Programa Recursivo

P é $\mathcal{R};\mathcal{S}$ onde
$\mathcal{R}$ def F;(se T então $\mathcal{R}$ senão $G;\mathcal{S}$),
$\mathcal{S}$ def (se T então ✓ senão $F;\mathcal{R}$)

Computação de um Programa Recursivo

- A **computação** de um programa recursivo consiste na **avaliação da expressão inicial**
- **Cada identificador de sub-rotina** referenciado é **substituído** pela correspondente **expressão** que o define
- A **recursão termina** quando um **identificador** de sub-rotina for **substituído pela operação vazia** ✓
- É possível representarmos um programa recursivo através de fluxogramas, assim como ocorre com programas monolíticos e iterativos?

Exercícios

1. Crie um programa monolítico com 6 instruções e, pelo menos, 2 testes. Descreva o programa utilizando instruções rotuladas

2. Apresente o fluxograma correspondente ao programa monolítico do exercício anterior

3. Crie um programa iterativo com 6 instruções e, pelo menos, uma instrução `enquanto` e uma `até`

4. Apresente o fluxograma correspondente ao programa iterativo do exercício anterior

5. Crie um programa recursivo com 3 expressões de sub-rotinas