Paradigmas de Linguagens de Programação

LISTA DE EXERCÍCIOS

Conceitos de Linguagens de Programação
Robert Sebesta, Bookman, 2010, 9ª Edição

CAPITULO 1

- Questões de Revisão (página 53): 1, 6, 7, 9, 13, 14, 20, 21, 24, 29 e 30
- Conjunto de Problemas (páginas 54 e 55): 2, 3, 4, 5, 12 e 17

CAPITULO 2

- Questões de Revisão (página 131 e 132): 27, 35, 58, 53
- Conjunto de Problemas (páginas 133 e 134): 7, 12, 17 e 24

CAPITULO 5

- Questões de Revisão (página 261): 1, 3, 11 e 17

CAPITULO 6

- Questões de Revisão (páginas 337 e 338): 2, 3, 12, 36, 37

CAPITULO 7

- Questões de Revisão (páginas 366 e 367): 7, 10, 16, 17, 18, 19 e 27

CAPÍTULO 8

- Questões de Revisão (páginas 337 e 338): 8, 12 e 18
- Exercícios de Programação: 4 e 5

CAPÍTULO 1

QUESTÕES DE REVISÃO

1. Por que é útil para um programador ter alguma experiência no projeto de linguagens, mesmo que ele nunca projete uma linguagem de programação?
2. Como o conhecimento de linguagens de programação pode beneficiar toda a comunidade da computação?
3. Que linguagem de programação tem dominado a computação científica nos últimos 50 anos?
4. Que linguagem de programação tem dominado as aplicações de negócios nos últimos 50 anos?
5. Que linguagem de programação tem dominado a Inteligência Artificial nos últimos 50 anos?
6. Em que linguagem o UNIX é escrito?
7. Qual é a desvantagem de ter muitas características em uma linguagem?
8. Como a sobrecarga de operador definida pelo usuário pode prejudicar a legibilidade de um programa?
9. Cite um exemplo da falta de ortogonalidade no projeto da linguagem C.
10. Qual linguagem usou a ortogonalidade como um critério de projeto primário?
11. Que sentença de controle primitiva é usada para construir sentenças de controle mais complicadas em linguagens que não as têm?
12. Que construção de uma linguagem de programação fornece abstração de processos?
13. O que significa para um programa ser confiável?
14. Por que verificar os tipos dos parâmetros de um subprograma é importante?
15. O que são apelidos?
16. O que é o tratamento de exceções?
17. Por que a legibilidade é importante para a facilidade de escrita?
18. Como o custo de compiladores para uma linguagem está relacionado ao projeto dela?

19. Qual tem sido a influência mais forte no projeto de linguagens de programação nos últimos 50 anos?
20. Qual é o nome da categoria de linguagens de programação cuja estrutura é dita-da pela arquitetura de computadores de von Neumann?
21. Que duas deficiências das linguagens de programação foram descobertas como um resultado da pesquisa em desenvolvimento de software dos anos 1970?
22. Quais são os três recursos fundamentais de uma linguagem orientada a objetos?
23. Qual foi a primeira linguagem a oferecer suporte aos três recursos fundamentais da programação orientada a objetos?
24. Dê um exemplo de dois critérios de projeto de linguagens que estão em conflito direto um com o outro.
25. Quais são os três métodos gerais de implementar uma linguagem de programação?
26. Qual produz uma execução de programas mais rápida, um compilador ou um interpretador puro?
27. Que papel a tabela de símbolos tem em um compilador?
28. O que faz um ligador?
29. Por que o gargalo de von Neumann é importante?
30. Quais são as vantagens de implementar uma linguagem com um interpretador puro?

CONJUNTO DE PROBLEMAS

1. Você acredita que nossa capacidade de abstração é influenciada por nosso domínio de linguagens? Defenda sua opinião.
2. Cite alguns dos recursos de linguagens de programação específicas que você conhece cujo objetivo seja um mistério para você.
3. Que argumentos você pode dar a favor da ideia de uma única linguagem para todos os domínios de programação?
4. Que argumentos você pode dar contra a ideia de uma única linguagem para todos os domínios de programação?
5. Nomeie e explique outro critério pelo qual as linguagens podem ser julgadas (além dos discutidos neste capítulo).
6. Que sentença comum das linguagens de programação, em sua opinião, é mais prejudicial à legibilidade?
7. Java usa um símbolo de fechamento de chaves para marcar o término de todas as sentenças compostas. Quais são os argumentos a favor e contra essa decisão de projeto?
8. Muitas linguagens distinguem entre letras minúsculas e maiúsculas em nomes definidos pelo usuário. Quais são as vantagens e desvantagens dessa decisão de projeto?
9. Explique os diferentes aspectos do custo de uma linguagem de programação.
10. Quais são os argumentos para escrever programas eficientes mesmo sabendo que os sistemas de hardware são relativamente baratos?
11. Descreva alguns *trade-offs* de projeto entre a eficiência e a segurança em alguma linguagem que você conheça.
12. Quais recursos principais uma linguagem de programação perfeita deveria incluir, em sua opinião?

13. A primeira linguagem de programação de alto nível que você aprendeu era implementada com um interpretador puro, um sistema de implementação híbrido ou um compilador? (Você não necessariamente saberá isso sem pesquisar).
14. Descreva as vantagens e desvantagens de alguns ambientes de programação que você já tenha usado.
15. Como sentenças de declaração de tipos para variáveis simples afetam a legibilidade de uma linguagem, considerando que algumas não precisam de tais declarações?
16. Escreva uma avaliação de alguma linguagem de programação que você conheça, usando os critérios descritos neste capítulo.
17. Algumas linguagens de programação – por exemplo, Pascal – têm usado o ponto e vírgula para separar sentenças, enquanto Java os utiliza para terminar sentenças. Qual desses usos, em sua opinião, é mais natural e menos provável de resultar em erros de sintaxe? Justifique sua resposta.
18. Muitas linguagens contemporâneas permitem dois tipos de comentários: um no qual os delimitadores são usados em ambas as extremidades (comentários de múltiplas linhas) e um no qual um delimitador marca apenas o início do comentário (comentário de uma linha). Discuta as vantagens e desvantagens de cada um dos tipos de acordo com nossos critérios.

CAPÍTULO 2

QUESTÕES DE REVISÃO

QUESTÕES DE REVISÃO

1. Em que ano Plankalkül foi projetada? Em que ano foi publicado o projeto?
2. Cite duas estruturas de dados comuns incluídas em Plankalkül.
3. Como eram implementados os pseudocódigos do início dos anos 1950?
4. *Speedcoding* foi inventada para resolver duas limitações significativas do hardware computacional do início dos anos 1950. Que limitações eram essas?
5. Por que a lentidão da interpretação dos programas era aceitável no início dos anos 1950?
6. Que recursos de hardware que apareceram pela primeira vez no computador IBM 704 afetaram fortemente a evolução das linguagens de programação. Explique por quê.
7. Em que ano foi iniciado o projeto do Fortran?
8. Qual era a área primária de aplicação dos computadores na época em que o Fortran foi projetado?
9. Qual foi a fonte de todas as sentenças de fluxo de controle do Fortran I?
10. Qual foi o recurso mais significativo adicionado ao Fortran I para chegar ao Fortran II?
11. Quais sentenças de controle de fluxo foram adicionadas ao Fortran IV para chegar ao Fortran 77?
12. Que versão do Fortran foi a primeira a ter quaisquer tipos de variáveis dinâmicas?
13. Que versão do Fortran foi a primeira a ter manipulação de cadeias de caracteres?
14. Por que os linguistas estavam interessados em inteligência artificial no final dos anos 1950?
15. Onde o LISP foi desenvolvido? Por quem?
16. De que maneira Scheme e COMMON LISP são linguagens opostas?
17. Que dialeto de LISP é usado para cursos introdutórios de programação em algumas universidades?
18. Quais são as duas organizações profissionais que projetaram o ALGOL 60?
19. Em que versão do ALGOL a estrutura de bloco apareceu?
20. Que elemento de linguagem que faltava ao ALGOL 60 fez com que suas chances de uso disseminado fossem prejudicadas?
21. Que linguagem foi projetada para descrever a do ALGOL 60?
22. Em que linguagem o COBOL foi baseado?
23. Em que ano o processo de projeto do COBOL começou?
24. Que estrutura de dados apareceu no COBOL que foi originada em Plankalkül?
25. Que organização foi a maior responsável pelo sucesso inicial do COBOL (em termos de uso)?
26. Para que grupo de usuários foi focada a primeira versão do BASIC?

27. Por que BASIC foi uma linguagem importante no início dos anos 1980?
28. PL/I foi projetada para substituir que duas outras linguagens?
29. Para que nova linha de computadores PL/I foi projetada?
30. Que recursos de SIMULA 67 são agora partes importantes de algumas linguagens orientadas a objetos?
31. Que inovação em estruturas de dados foi introduzida no ALGOL 68, geralmente creditada ao Pascal?
32. Que critério de projeto foi usado extensivamente em ALGOL 68?
33. Que linguagem introduziu a sentença `case`?
34. Que operadores em C foram modelados a partir de operadores similares em ALGOL 68?
35. Cite duas características de C que o tornam menos seguro do que Pascal.
36. O que é uma linguagem não procedural?
37. Quais são os dois tipos de sentenças que compõem uma base de dados Prolog?
38. Qual é a área de aplicação primária para a qual Ada foi projetada?
39. Como são chamadas as unidades de programas concorrentes em Ada?
40. Que construção de Ada fornece suporte para tipos abstratos de dados?
41. O que compõe o mundo de Smalltalk?
42. Quais são os três conceitos base para a programação orientada a objetos?
43. Por que C++ inclui os recursos de C que são sabidamente inseguros?
44. O que as linguagens Ada e COBOL têm em comum?
45. Qual foi a primeira aplicação para Java?
46. Que característica de Java é mais evidente em JavaScript?
47. Como o sistema de tipos de PHP e JavaScript diferem daquele de Java?
48. Que estrutura de vetor é incluída em C#, mas não em C, C++ ou Java?
49. Quais são as duas linguagens que a versão original de Perl pretendia substituir?
50. Para qual área de aplicação JavaScript é mais usada?
51. Qual é o relacionamento entre JavaScript e PHP, em termos de utilização?
52. A estrutura de dados primária de PHP é uma combinação de que duas outras estruturas de dados de outras linguagens?
53. Que estrutura de dados Python usa em vez de vetores?
54. Que características Ruby compartilha com Smalltalk?
55. Que característica dos operadores aritméticos de Ruby os tornam únicos entre aqueles de outras linguagens?
56. Que estruturas de dados são construídas em Lua?
57. Lua é normalmente compilada, puramente interpretada ou impuramente interpretada?
58. Que recurso das classes do Delphi é incluído em C#?
59. Que deficiência da sentença `switch` do C é sanada com as mudanças feitas por C# a essa construção?
60. Qual é a plataforma primária na qual o C# é usado?
61. Quais são as entradas para um processador XSLT?
62. Qual é a saída de um processador XSLT?
63. Que elemento da JSTL é relacionado a um subprograma?
64. Por que um documento JSP é convertido por um processador JSP?
65. Os *servlets* são executados?

CONJUNTO DE PROBLEMAS

1. Que recursos de Plankalkül você acha que teriam maior influência no Fortran 0 se os projetistas do Fortran estivessem familiarizados com Plankalkül?
2. Determine as capacidades do sistema 701 Speedcoding de Backus e compare-as com as de uma calculadora de mão programável.
3. Escreva uma breve história dos sistemas A-0, A-1 e A-2 projetados por Grace Hooper e seus associados.
4. Como um projeto de pesquisa, compare as facilidades do Fortran 0 com as do sistema de Laning e Zierler.
5. Qual dos três objetivos originais do comitê de projeto do ALGOL, na sua opinião, foi mais difícil de ser atingido naquela época?
6. Em sua opinião, qual é o erro de sintaxe mais comum em programas LISP?
7. LISP começou como uma linguagem funcional pura, mas gradualmente foi adquirindo mais recursos imperativos. Por quê?
8. Descreva em detalhes as três razões mais importantes, na sua opinião, por que o ALGOL 60 não se tornou uma linguagem amplamente usada.
9. Por que, na sua opinião, o COBOL permite identificadores longos, enquanto Fortran e ALGOL não permitiam?
10. Descreva a maior motivação da IBM para desenvolver PL/I.
11. Era correta a interpretação da IBM na qual foi baseada sua decisão para desenvolver PL/I, dada a história dos computadores e os desenvolvimentos de linguagem desde 1964?
12. Descreva, em suas próprias palavras, o conceito de ortogonalidade no projeto de linguagens de programação.
13. Qual é a razão primária pela qual PL/I se tornou mais usada do que o ALGOL 68?
14. Quais são os argumentos a favor e contra a ideia de uma linguagem sem tipos?
15. Existem outras linguagens de programação lógica além de Prolog?
16. Qual a sua opinião sobre do argumento de que as linguagens muito complexas também são muito perigosas, e que devemos manter todas as linguagens pequenas e simples?
17. Você pensa que o projeto de linguagem por comitê é uma boa ideia? Justifique sua resposta.
18. As linguagens continuam a evoluir. Que tipo de restrições você acha adequadas para mudanças em linguagens de programação? Compare suas respostas com a evolução do Fortran.
19. Construa uma tabela identificando todas as principais evoluções das linguagens, constando quando elas ocorreram, em quais linguagens apareceram primeiro e as identidades dos desenvolvedores.
20. Existiram algumas trocas públicas entre a Microsoft e a Sun a respeito do projeto do J++ e do C# da Microsoft e do Java da Sun. Leia alguns desses documentos, disponíveis em seus respectivos sites Web, e escreva uma análise das discordâncias existentes.
21. As linguagens de *scripting* têm evoluído as estruturas de dados de forma a substituir os vetores tradicionais. Explique a sequência cronológica desses avanços.
22. Dê duas razões para que a interpretação pura seja um método de implementação aceitável para diversas das linguagens de *scripting* recentes.

23. Perl 6, quando chegar, provavelmente será uma linguagem significativamente ampliada. Tente estimar se uma linguagem como Lua também crescerá continuamente ao longo de seu tempo de vida. Justifique sua resposta.
24. Por que, em sua opinião, aparecem novas linguagens de *scripting* mais frequentemente do que novas linguagens compiladas?
25. Dê uma breve descrição geral de uma linguagem híbrida de marcação/programação.

EXERCÍCIOS DE PROGRAMAÇÃO

1. Para entender o valor dos registros em uma linguagem de programação, escreva um pequeno programa em uma linguagem baseada em C que use um vetor de estruturas que armazenem informações de estudantes, incluindo o nome, a idade, a média das notas como um ponto-flutuante e o nível do estudante em uma cadeia (por exemplo, "calouro" etc.). Escreva também o mesmo programa na mesma linguagem sem usar tais estruturas.
2. Para entender o valor da recursão em uma linguagem de programação, escreva um programa que implemente o algoritmo *quicksort*, primeiro usando recursão e então sem usar recursão.
3. Para entender o valor dos laços de iteração de contagem, escreva um programa que implemente multiplicação de matrizes usando construções de repetição baseadas em contagem. Então, escreva o mesmo programa usando apenas laços de repetição lógicos – por exemplo, laços `while`.

CAPÍTULO 5

QUESTÕES DE REVISÃO

1. Quais são as questões de projeto para nomes?
2. Qual é o perigo em potencial dos nomes sensíveis a capitalização?
3. De que forma as palavras reservadas são melhores do que as palavras-chave?
4. O que é um apelido?
5. Que categoria de variáveis de referência em C++ é sempre composta de apelidos?
6. O que é o lado esquerdo de uma variável? O que é o lado direito?
7. Defina *vinculação* e *tempo de vinculação*.
8. Após o projeto e implementação de uma linguagem, quais são os quatro tipos de vinculações que podem ocorrer em um programa?
9. Defina *vinculação estática* e *vinculação dinâmica*.
10. Quais são as vantagens e desvantagens de declarações implícitas?
11. Quais são as vantagens e desvantagens da vinculação de tipos dinâmica?
12. Defina *variáveis estáticas*, *dinâmicas da pilha*, *dinâmicas do monte explícitas* e *dinâmicas do monte implícitas*.
13. Defina *tempo de vida*, *escopo*, *escopo estático* e *escopo dinâmico*.
14. Como a referência a uma variável não local em um programa com escopo estático está conectada a sua definição?
15. Qual é o problema geral do escopo estático?
16. O que é o ambiente de referenciamento de uma sentença?
17. O que é um ancestral estático de um subprograma? O que é um ancestral dinâmico de um subprograma?
18. O que é um bloco?
19. Quais são as vantagens e as desvantagens do escopo dinâmico?
20. Quais são as vantagens das constantes nomeadas?

CAPÍTULO 6

QUESTÕES DE REVISÃO

1. O que é um descritor?
2. Quais são as vantagens e as desvantagens dos tipos de dados decimais?
3. Quais são as questões de projeto para tipos de cadeias de caracteres?
4. Descreva as três opções para tamanhos de cadeias.
5. Defina tipos *ordinais*, de *enumeração* e de *subfaixa*.
6. Quais são as vantagens dos tipos de enumeração definidos pelo usuário?
7. De que maneira os tipos de enumeração definidos pelo usuário de C# são mais confiáveis do que os de C++?
8. Quais são as questões de projeto para matrizes?
9. Defina matrizes *estáticas*, *dinâmicas da pilha fixas*, *dinâmicas da pilha*, *dinâmicas do monte fixas* e *dinâmicas do monte*. Quais são as vantagens de cada uma delas?
10. O que acontece quando um elemento não existente de uma matriz é referenciado em Perl?
11. Como JavaScript suporta matrizes esparsas?
12. Que linguagens suportam índices negativos?
13. Que linguagens suportam fatias de matrizes com tamanhos de passos (*stepsizes*)?
14. Que recurso de inicialização de matrizes está disponível em Ada e não está disponível em outras linguagens imperativas comuns?
15. O que é uma constante agregada?

16. Que operações de matrizes são fornecidas especificamente para matrizes unidimensionais em Ada?
17. Quais são as diferenças entre as fatias do Fortran 95 e as de Ada?
18. Defina *ordem principal de linha* e *ordem principal de coluna*.
19. O que é uma função de acesso para uma matriz?
20. Quais são as entradas requeridas em um descritor de matriz em Java e quando elas devem ser armazenadas (em tempo de compilação ou em tempo de execução)?
21. Qual é o propósito dos números de níveis em registros no COBOL?
22. Defina *referências completamente qualificadas* e *elípticas* para campos em registros.
23. Defina *união*, *união livre* e *união discriminada*.
24. Quais são as questões de projeto para uniões?
25. As uniões de Ada são sempre verificadas em relação ao tipo?
26. Quais são as questões de projeto para os tipos ponteiro?
27. Quais são os dois problemas comuns com ponteiros?
28. Por que os ponteiros da maioria das linguagens são restritos de forma que apontem uma única variável de tipo?
29. O que é um tipo de referência em C++ e para que ele é comumente usado?
30. Por que as variáveis de referência em C++ são melhores do que os ponteiros para parâmetros formais?
31. Quais vantagens as variáveis de tipo de referência em Java e C# têm em relação aos ponteiros em outras linguagens?
32. Descreva a abordagem preguiçosa e a ansiosa para recuperar lixo.
33. Por que a aritmética de referências em Java e C# não faz sentido?
34. O que é um tipo compatível?
35. Defina erro de tipo.
36. Defina fortemente tipada.
37. Por que Java não é fortemente tipada?
38. O que é uma conversão implícita sem conversão?
39. Por que C e C++ não são fortemente tipadas?
40. O que é a equivalência de tipos por nome?
41. O que é a equivalência de tipos por estrutura?
42. Qual é a vantagem principal da equivalência de tipos por nome?
43. Qual é a desvantagem principal da equivalência de tipos por estrutura?
44. Para que tipos o C usa a equivalência de tipos por estrutura?
45. Que operações de conjunto modelam o tipo de dados `struct` de C?

CAPÍTULO 7

QUESTÕES DE REVISÃO

1. Defina *precedência de operador* e *associatividade de operador*.
2. O que é um operador ternário?
3. O que é um operador pré-fixado?
4. Que operador normalmente tem associatividade à direita?

5. O que é um operador não associativo?
6. Que regras de associatividade são usadas por APL?
7. Qual é a diferença entre a maneira pela qual os operadores são implementados em C++ e Ruby?
8. Defina *efeito colateral funcional*.
9. O que é uma coerção?
10. O que é uma expressão condicional?
11. O que é um operador sobrecarregado?
12. Defina *conversões de alargamento e de estreitamento*.
13. Qual é a diferença entre == e === em JavaScript?
14. O que é uma expressão de modo misto?
15. O que é transparência referencial?
16. Quais são as vantagens da transparência referencial?
17. Como a ordem de avaliação dos operandos interage com os efeitos colaterais funcionais?
18. O que é a avaliação em curto-circuito?
19. Diga uma linguagem que sempre faz avaliação em curto-circuito de expressões booleanas. Diga uma que nunca faz isso. Diga uma na qual o programador pode escolher.
20. Como C oferece suporte para expressões relacionais e booleanas?
21. Qual é o propósito de um operador de atribuição composto?
22. Qual é a associatividade dos operadores aritméticos unários de C?
23. Qual é uma possível desvantagem de tratar o operador de atribuição como se ele fosse um operador aritmético?
24. Que duas linguagens incluem atribuições de listas?
25. Que atribuições de modo misto são permitidas em Ada?
26. Que atribuições de modo misto são permitidas em Java?
27. O que é um *cast*?

CAPÍTULO 8

QUESTÕES DE REVISÃO

15. O que é uma sentença de laço com pré-teste? O que é uma sentença de laço com pós-teste?
16. Explique como a sentença `Do` do Fortran funciona.
17. Qual é a diferença entre a sentença **`for`** de C++ e a de Java?
18. De que maneira a sentença **`for`** do C é mais flexível do que a de muitas outras linguagens?
19. Qual é o escopo de uma variável de laço em Ada?
20. O que a função **`range`** faz em Python?
21. Cite uma diferença fundamental entre o **`for`** numérico de Lua e o **`Do`** do Fortran.
22. Que linguagens contemporâneas não incluem um `goto`?
23. Quais são as questões de projeto para sentenças de laços controlados logicamente?
24. Qual é a principal razão para a invenção das sentenças de controle de laço posicionadas pelo usuário?
25. Quais são as questões de projeto para mecanismos de controle de laço posicionados pelo usuário?
26. Qual a vantagem da sentença **`break`** de Java em relação à sentença **`break`** do C?
27. Quais são as diferenças entre a sentença **`break`** de C++ e a de Java?
28. O que é um controle de iteração definido pelo usuário?
29. Que linguagens de programação bastante usadas pegam emprestado parte de seu projeto dos comandos protegidos de Dijkstra?

EXERCÍCIOS DE PROGRAMAÇÃO