01. [Sebesta, 2000] Qual é a definição de estrutura de controle?

02. [Sebesta, 2000] Qual é a definição de bloco?

03. [Sebesta, 2000] Quais são as questões de projeto relativas às estruturas de seleção?

04. [Sebesta, 2000] Quais são as soluções comuns para o problema do aninhamento de seletores bidirecionais? O que está errado com a solução do Modula-2?

05. [Sebesta, 2000] Quais são as questões de projeto referentes às instruções de seleção múltipla?

06. [Sebesta, 2000] Qual é a base para as instruções de controle do FORTRAN I?

07. [Sebesta, 2000] O que está errado com a instrução `IF` aritmética do FORTRAN?

08. [Sebesta, 2000] O que é incomum a respeito da instrução de seleção múltipla do C? Qual *trade-off* de projeto foi feito?

09. [Sebesta, 2000] Quais são as questões de projeto referentes às instruções de laço controladas por contador?

10. [Sebesta, 2000] O que é uma instrução de laço pré-teste? E uma instrução pós-teste?

11. [Sebesta, 2000] Qual é a mudança mais significativa no *design* da instrução `DO` entre o FORTRAN IV e o FORTRAN 77?

12. [Sebesta, 2000] Qual característica da instrução `for` do ALGOL 60 torna os programas que a usam difíceis de serem lidos?

13. [Sebesta, 2000] Qual é a diferença entre a instrução `for` do C++ e a do Java?

14. [Sebesta, 2000] Quais são as questões de projeto referentes às instruções de laço controladas logicamente?

15. [Sebesta, 2000] Qual é a principal razão para que instruções de controle de laço localizado pelo usuário tenham sido inventadas?

16. [Sebesta, 2000] Que vantagem a instrução `exit` da Ada tem sobre a instrução `break` do C?

17. [Sebesta, 2000] Quais são as diferenças entre a instrução `break` do C++ e a do Java?

18. [Sebesta, 2000] O que é um controle de iteração definido pelo usuário?

19. [Sebesta, 2000] Quais são as duas desvantagens das variáveis de rótulos da PL/I?

20. [Sebesta, 2000] Qual é o principal problema com as restrições `goto` do Pascal?

21. [Sebesta, 2000] Quais linguagens de programação comuns tomam emprestado parte de seu projeto dos comandos protegidos de Dijkstra?

22. [Sebesta, 2000] Redija uma defesa da afirmação segundo a qual a instrução de seleção tridirecional do FORTRAN I foi a melhor opção, dadas as circunstâncias da época.

23. [Sebesta, 2000] Imagine uma situação na qual a variável de rótulo da PL/I seria uma grande vantagem.

24. [Sebesta, 2000] Descreva três situações em que uma construção de laço de contagem e lógico combinados seja necessária.

25. [Sebesta, 2000] Compare o `GO TO` computada do FORTRAN com a instrução `case` Pascal, especialmente em termos de legibilidade e de confiabilidade.

26. [Sebesta, 2000] Quais são as possíveis razões para que o Pascal tenha um *loop* pós-teste lógico, enquanto o ALGOL 60 não?

27. [Sebesta, 2000] Estude o recurso iterador da CLU em Liskov *et al.* (1984) e determine suas vantagens e suas desvantagens.

28. [Sebesta, 2000] Compare o conjunto de instruções de controle da Ada com as do FORTRAN 77 e decida qual é melhor e por quê.

29. [Sebesta, 2000] Quais são os prós e os contras de usar palavras reservadas de fechamento únicas em instruções compostas?

30. [Sebesta, 2000] Analise os potenciais problemas de legibilidade que existem no uso de palavras reservadas de fechamento para instruções de controle que são o inverso das palavras reservadas iniciais correspondentes, como por exemplo, as palavras reservadas `case-esac` do ALGOL 68. Por exemplo, considere erros de digitação comuns, como a inversão de dois caracteres adjacentes.

31. [Sebesta, 2000] Reescreva o seguinte segmento de código usando uma estrutura de laço nas seguintes linguagens:

```
  k := (j + 13) / 27
loop:
  if k > 10 then goto out
  k := k + 1
  i := 3 * k - 1
  goto loop
out: ...
```

a) Pascal

b) FORTRAN 77

c) Ada

d) C, C++ ou Java

Suponha que todas as variáveis sejam do tipo inteiro. Discuta qual linguagem, para esse código, tem a melhor capacidade de escrita, a melhor legibilidade e a melhor combinação das duas.

32. [Sebesta, 2000] Refaça o Problema 31, exceto que, desta vez, transforme todas as variáveis e constantes no tipo ponto-flutuante e mude a instrução `k := k + 1` para `k := k + 1.2`

33. [Sebesta, 2000] Reescreva o seguinte segmento de código usando uma instrução de seleção múltipla nas seguintes linguagens:

```
if (k = 1) or (k = 2) then j := 2 * k - 1
if (k = 3) or (k = 5) then j := 3 * k + 1
if (k = 4) then j := 4 * k - 1
if (k = 6) or (k = 7) or (k = 8) then j := k - 2
```

a) Pascal

b) FORTRAN 90 (você terá de pesquisar esta)

c) Ada

d) C, C++ ou Java

Suponha que todas as variáveis sejam do tipo inteiro. Discuta os méritos relativos do uso dessas linguagens para esse código particular.

34. [Sebesta, 2000] Considere a seguinte instrução `for` ao estilo ALGOL 60:

```
for i := j + 1 step i * j until 3 * j do j := j + 1
```

Suponha que o valor de `j` seja `1`. Liste a sequência de valores para a variável `i` usada, supondo a seguinte semântica:

a) Todas as expressões são avaliadas na entrada do laço.

b) Todas as expressões são avaliadas antes de cada iteração.

c) Expressões `step` são avaliadas uma vez na entrada do laço, e expressões `until` são avaliadas antes de cada iteração.

d) Expressões `until` são avaliadas uma vez na entrada do laço, e expressões `step` são avaliadas antes de cada iteração, logo depois que o contador de laços é incrementado.

Em todos os casos, quando mais de uma expressão é avaliada ao mesmo tempo, elas são avaliadas na ordem esquerda para a direita. Além disso, a atribuição é sempre feita assim que seu RHS é avaliado.

35. [Sebesta, 2000] Use o *Science Index* para encontrar um artigo que se refira a Knuth (1974). Leia o artigo e o documento de Knuth e redija um documento que resuma ambos os lados do argumento `goto`.

36. [Sebesta, 2000] Nesse documento sobre a questão da `goto`, Knuth (1974) sugere uma construção de controle de laço que permite saídas múltiplas. Leia o documento e redija uma descrição semântica operacional da construção.

37. [Sebesta, 2000] Considere a seguinte instrução `case` Pascal. Reescreva-a usando somente a seleção bidirecional

```
case indice - 1 of
  2, 4 : par := par + 1;
  1, 3 : impar := impar + 1;
  0 : zero := zero + 1;
  else erro := true
end
```

38. [Sebesta, 2000] Considere o seguinte segmento de programa C. Reescreva-o sem usar nenhum `goto` ou `break`:

```
j = -3;
for(i = 0; i < 3; i++) {
  switch(j + 2) {
    case 3:
    case 2: j--; break;
    case 0: j += 2; break;
    default: j = 0;
  }
  if(j > 0) break;
  j = 3 - i;
}
```

39. [Sebesta, 2000] Em uma carta ao editor da CACM, Rubin (1987) usa o seguinte segmento de código como prova de que a legibilidade de códigos com `goto` é melhor do que um código equivalente sem `goto`. Esse código localiza a primeira linha de uma matriz de números inteiros *n x n* chamada `x` que nada tem a não ser valores zero.

```
for i := 1 to n do
begin
  for j := 1 to n do
    if x[i, j] <> 0
      then goto rejeita;
    writeln('Primeira linha só com zeros é:', i);
    break;
  rejeita:
end;
```

Reescreva esse código sem `goto` em uma das seguintes linguagens: C, C++, Pascal, Java ou Ada. Compare a legibilidade de seu código com a do código acima.

40. [Sebesta, 2000] Quais são os argumentos (a favor e contra) com relação ao uso exclusivo de expressões booleanas nas instruções de controle em Java (em oposição a também permitir expressões aritméticas, como no C e no C++)?