Segunda Lista de Exercícios de Linguagens de Programação.
Primeiro Semestre de 2007. Dept. de Computação – UFSCar.
José de Oliveira Guimarães.

1. O que é erro de tipos ?

2. Que características possui uma linguagem fortemente tipada ? E uma estaticamente tipada ?

3. Dê um exemplo de um programa em S que utilize procedimentos encaixados. O programa deve funcionar, mesmo que não faça nada de útil.

4. Cite as desvantagens de estruturas de blocos.

5. Faça um pequeno programa com dois procedimentos na linguagem S com escopo dinâmico de tal forma que :

- não haja erro de compilação nem de execução;
- se a linguagem suportasse apenas escopo estático, haveria um erro em tempo de compilação.

6. Admita que o programa abaixo seja de uma linguagem com escopo dinâmico. O que ele escreve ? Naturalmente, a execução do programa começa no procedimento `main`.

```
proc P()
  var i, n : integer;
  begin
  n = k + r;
  for i = 1 to n do
    max = max + i;
  end

proc Q( max : integer )
  var k : integer;
  begin
  k = 1;
  P();
  writeln(max);
  k = 3;
  P();
  writeln(max);
  writeln(n);
  end


proc main()
```

```
  var n, r : integer;
  begin
  n = 5;
  r = 1;
  Q(n);
  end
```

7. O que escreve o programa abaixo ? O que acontece se a constante `max` for inicializada com `1` ?

```
    { variaveis globais }
  var a, b;

proc P( i )
  begin
  if i > 3
  then
    a = 12;
    b = "abcdefg";
  else
    a = "abcdefg";
    b = 33;
  endif
  end

proc main()
  var s, j;
  begin
  max = 5;
  s = 1;
  for j = 1 to max do
    s = s + j;
  P(s);
  a = a + 1;
  writeln(a);
  end
```

8. Explique como módulos funcionam. A sua resposta deve explicar o que é parte pública e privada e o que um módulo pode utilizar de outro módulo que ele importa.

9. Cite duas vantagens de se utilizar módulos.

10. Faça um pequeno exemplo com dois procedimentos de tal forma que uma exceção levantada em um deles seja tratada no outro. Utilizando este exemplo, explique como exceções funcionam.

11. Faça um exemplo onde uma exceção é levantada e não tratada.

12. Cite duas desvantagens de exceções.

13. Que tipos de erros podem acontecer quando o programador desaloca memória explicitamente ?

14. Cite quatro problemas com a desalocação explícita de memória pelo programador (`delete` de C++, `dispose` de Pascal).

15. Cite dois problemas com coleta de lixo (mesmo se os coletores atuais já resolveram estes problemas, pelo menos parcialmente).

16. Defina objeto. É um objeto um valor ? Isto é, ele se assemelha mais ao valor 5 que está um `i` após a instrução

```
i = 5;
```

ser executada ou ele se assemelha mais ao:

- tipo `integer`;
- variável `i`;

?

17. É um objeto mais semelhante a uma pessoa ou ao seu nome ?

18. É um objeto mais semelhante a um projeto de um avião 767 ou a um avião que existe no mundo real ?

19. Explique as vantagens de proteção de informação usando um exemplo.

20. Explique porque proteção de informação impede que modificações na representação (variáveis de instância) de uma classe invalidem outras classes do mesmo programa.

21. Proteção de informação faz os programas se tornarem mais rápidos ? Explique.

22. Subclasses são mais gerais ou específicas do que as superclasses ?

23. Uma subclasse pode redefinir um método herdado da superclasse ? Mostre um exemplo.

24. Faça um exemplo de classe onde a palavra chave `super` é utilizada.

25. Explique detalhadamente, através de um exemplo, como polimorfismo causa reuso de código.

26. Uma classe `Desenho` possui um vetor de objetos de `Figura` (classe dada em aula). Um dos métodos desta classe é dado abaixo.

```
class Desenho
  private:
    var v : array(Figura)[];
        size_v : integer;
    ...
  public:
    proc desenhe()
```

```
        var i : integer;
      begin
      for i = 0 to n do
        v[i].desenhe();
      end
    ...
endclass
```

Um objeto da classe `Desenho` contém referencias para objetos de `Figura` e suas subclasses `Retangulo`, `Triangulo`, `Circulo`, `Elipse` e `Poligono`. Pergunta-se: esta classe poderia ser feita declarando-se o vetor como

```
var v : array(Circulo)[];
```

? Responda explicando se `v` poderia se referir aos objetos de `Figura` e suas subclasses.

27. Baseado na hierarquia

```
class A
  public:
    proc s() begin end
endclass

class B inherits A
  public:
    proc m() begin end
endclass

class C inherits B
  public:
    proc s() begin end
    proc m() begin end
endclass
```

responda quais métodos serão executados pelos comandos abaixo. Alguns dos comandos resultam em erros de compilação. Quando isto acontecer, ignore o comando.

```
proc main()
  var a : A;
      b : B;
      c : C;
begin
a = A.new();
a.s();
a.m();

b = B.new();
a = b;
a.s();
a.m();
b.s();
```

```
b.m();

c = C.new();
a = c;
a.s();
a.m();
c.s();
c.m();

b = c;
b.s();
b.m();

end
```

28. Utilizando a hierarquia do exercício anterior, cite os métodos de cada classe, inclusive os herdados. Cite também a procedência de cada método, como no exemplo:

```
classe F:
   F::m
   G::p
   F::h
```

onde `F` herda o método `p` de `G` e define os métodos `m` e `h`.

29. Descreva em palavras o funcionamento da classe `Politico`.

```
class Pessoa
  public:
    proc facaAlgumaCoisa() begin end
endclass

class Preguicoso inherits Pessoa
  public:
    proc facaAlgumaCoisa()
      begin
      write("naooooooo!!!!!!!\n");
      end
endclass

class Trabalhador inherits Pessoa
  public:
    proc facaAlgumaCoisa()
      begin
      write("Ja vou !\n");
      end
endclass

class Politico inherits Pessoa
  public:
```

```
    proc init()
        { inicializa o objeto.
          Sempre deve ser
          chamado apos a
          alocacao }
      begin
        { nasce preguicoso }
      preguicoso = Preguicoso.new();
      corrente = preguicoso;
      trabalhador = Trabalhador.new();
      end
    proc facaAlgumaCoisa()
      begin
      corrente.facaAlgumaCoisa();
      end
    proc emEpocaDeEleicao()
      begin
      corrente = trabalhador;
      end
    proc foraDeEpocaDeEleicao()
      begin
      corrente = preguicoso;
      end
  private:
    var corrente, trabalhador,
        preguicoso : Pessoa;
endclass
```

30. Empregando a definição da linguagem POOL-I, qual o tipo da classe `Politico` ? E da classe `Trabalhador` ?

31. Defina *subtipo* como este conceito é empregado por POOL-I.

32. Por que a definição de subtipo de POOL-I é mais abrangente do que a de subclasse de C++ ? Cite um exemplo.

33. Faça um programa correto em POOL-I que esteja incorreto em C++.

34. Dada a classe em Smalltalk

```
class Store
  private:
    var i;
  public:
    proc get()
      begin
      return i;
      end
```

```
    proc put( p_i )
      begin
      i = p_i;
      end
endclass
```

É necessário transformá-la em classe parametrizada ? Se sim, diga como. Se não, explique.

35. Compare os modelos de linguagens Smalltalk, POOL-I, C++. Qual(is) deles oferece mais reaproveitamento de *software* ? Qual(is) oferece mais segurança contra erros de tipos ?

36. Faça um programa que resultaria em um erro de compilação (erro de tipo) em POOL-I. O programa equivalente em Smalltalk deve causar erro em execução.

37. Faça um programa que resultaria em um erro de compilação (erro de tipo) em POOL-I. O programa equivalente em Smalltalk **não** deve causar erro em execução.

38. Utilizando as classes da Figura 1, escreva quais os métodos serão executados pelo código em POOL-I abaixo.

```
  var a : A;
      b : B;
begin
a = A.new();
b = B.new();
b.set(a);
b.put(12);
b.print();
a.print();
a.put(5);
a = b;
a.print();
end
```

Este código estaria correto se a linguagem fosse C++ ? Explique.

39. Utilizando as classes do exercício anterior, o que aconteceria se a linha

```
     b.set(a)
```

fosse substituída por

```
     b.set(b)
```

?

40. Que desvantagens possui o sistema de tipos de Java em relação ao de POOL-I ? E que vantagens possui o sistema de tipos de Java em relação a C++ ?

41. Considerando-se apenas o sistema de tipos, os modelos estudados, pode-se transformar facilmente qualquer programa em Java em C++ ? E o contrário ? Justifique.

42. Considerando-se apenas o sistema de tipos, os modelos estudados, pode-se transformar facilmente qualquer programa em Java em POOL-I ? E o contrário ? Justifique.

```
class A
  private:
    var k : integer;
  public:
    proc put( p_k : integer )
      begin
      k = p_k;
      end
    proc print()
      begin
      write(k);
      end
endclass

class B
  private:
    var a : A;
  public:
    proc put( k : integer )
      begin
      a.put(k);
      end
    proc print()
      begin
      write(0);
      a.print();
      end
    proc set( p_a : A )
      begin
      a = p_a;
      end
endclass
```

Figura 1: Classes em POOL-I e C++

43. Considerando-se apenas o sistema de tipos, os modelos estudados, pode-se transformar facilmente qualquer programa em Smalltalk em POOL-I ? E o contrário ? Justifique.

44. Defina instanciação de uma classe parametrizada. Quando ela ocorre ? O código dos métodos da classe parametrizada é duplicada em cada instanciação ? Por quê ?

45. Faça uma classe parametrizada `Vetor` que tenha pelo menos os métodos

```
proc get( i : integer ) : T
proc put( elem : T; i : integer )
```

46. Defina identidade de objetos. Dê um exemplo.

47. Por que persistência de objetos é importante ? Explique, em poucas linhas, como funcionam os métodos `store` e `retrieve`.

48. Cite exemplos de classes onde a definição de um iterador é útil. Que características em comum possuem estas classes ?