1ª Lista de exercícios de Paradigmas de Programação

Prof. Glauber Cintra – Entrega: 18/mar/2013

Engenharia de computação **Semestre 4 – 2012.2**
Alessandra Lino
Álisson Rangel
Gracyane Oliveira

1) Como o conhecimento das características das linguagens de programação pode beneficiar toda a comunidade de computação?

Tão importante quanto se saber resolver determinadas instâncias de um determinado problema é ter o discernimento de qual linguagem de programação é apropriada para cada situação a ser resolvida, ou seja, escolher aquela que mais possui recursos úteis, se há gasto de memória excessivo ou não, se o custo é baixo ou alto, etc.

Se como profissionais pudermos avaliar estes itens e escolhermos a linguagem apropriada para o problema no qual queremos uma solução, é mais alta a possibilidade de que não ocorram problemas como falta de recursos, consumo excessivo de memória etc.

2) No contexto das linguagens de programação, defina o que é legibilidade e indique por que ela é importante.

Refere-se a facilidade de compreensão do código, ou seja, a facilidade que se tem de entender a lógica existente no programa. Sua importância advém da facilitação no ato de programar, pois tendo uma boa legibilidade tal ato torna-se facilitado, bem como a manutenção em si.

3) Qual a desvantagem de haver recursos demasiados em uma linguagem de programação?

O excesso de recursos pode vir a afetar a *legibilidade* do código, deixando-o extenso e com recursos que provavelmente programadores com mais experiência na linguagem de programação usada, possa compreender a lógica computacional do código feito.

4) Cite uma linguagem voltada para aplicações científicas, outra voltada para aplicações comerciais e mais uma voltada para inteligência artificial.

C, Java e Prolog, respectivamente.

5) Qual o nome da categoria de linguagens de programação cuja estrutura é fortemente influenciada pela arquitetura de computador de Von Neumann?

O paradigma imperativo é influenciado pela arquitetura de Von Neumann. Nessa arquitetura há grande necessidade de comunicação entre a memória e o processador, ou seja, nas linguagens imperativas as variáveis tem papel preponderante.

6) Qual a principal influência no projeto de linguagens de programação nos últimos 50 anos?

A arquitetura básica dos computadores exerceu um efeito crucial sobre o projeto das linguagens. A maioria das mais populares dos últimos 50 anos foi projetada em função da arquitetura de computador prevalecente, chamada arquitetura de Von Neumann, a fim de

existir compatibilidade das linguagens de programação em computadores de fabricantes diferentes.

7) Quais argumentos você poderia utilizar contra a idéia de uma única linguagem para todos os domínios de programação?

Uma linguagem que tenha todos os domínios de programação pode ser: lenta, ter custo alto, legibilidade e confiabilidade baixas. Para se construir uma linguagem dessa natureza, provavelmente, leva-se mais tempo e mais recursos financeiros. Há também grande possibilidade de existirem trechos dentro da linguagem mal implementados por serem gerais demais e, assim, podem afetar alguma aplicação específica. Resumindo, dificilmente uma linguagem de fins generalizados terá o mesmo desempenho de uma linguagem criada para um fim específico.

8) O que produz uma execução mais rápida: um compilador ou um interpretador? Justifique.

Um compilador produz uma execução mais rápida, pois depois de compilado o programa, o código executável criado roda direto no computador quando acionado. Já nas linguagens interpretadas sempre é preciso a utilização do interpretador junto com o código-fonte para executar o programa, deixando a aplicação mais lenta.

9) O que faz um *linkeditor*?

Um linkeditor faz o processo de coletar programas de sistema e ligá-los aos programas de usuário, além de ligar também a outros programas de usuários previamente compilados que residem em bibliotecas.

10) Em que categoria de linguagens Lisp pode ser classificada e qual domínio de aplicação onde Lisp é mais utilizada?

Lisp é uma linguagem do paradigma funcional, onde a computação é levada a efeito através de cálculo de função. Lisp é mais utilizada na área de Inteligência Artificial.

11) No contexto das linguagens de programação, o que são variáveis?

São abstrações de endereços de memória, estes endereços guardam valores, conhecidos como *estado interno* das variáveis.

12) Quais são duas características que toda variável possui?

Possui um **identificador**, conhecido como *nome* e também possui associação a um **endereço de memória**, o conteúdo dessa região é chamado de *estado interno* da variável.

13) O que é vinculação dinâmica de memória? E vinculação estática?

Na **vinculação dinâmica**, a alocação de memória ocorre durante a *execução* do programa, ou seja, não há garantias que exista espaço de memória disponível. Já na **vinculação estática**, que é realizada durante a *compilação* do programa, há mais garantias de que exista espaço disponível para que o código execute suas funções. No entanto, uma variável que é alocada estaticamente não poderá alocar mais espaço ao longo do programa.

14) Diferencie variáveis escalares de variáveis compostas, variáveis estáticas de variáveis dinâmicas e variáveis globais de variáveis locais.

Variáveis **escalares** são as que armazenam um *valor de cada vez*. Como por exemplo:

```
int num = 3; //Variável escalar na linguagem C
```

Já variáveis **compostas** são aquelas que tem a capacidade de armazenar *uma coleção de valores* ao mesmo tempo. Como por exemplo:

```
int numeros [4]= {0, 1, 2, 3}; /*Vetor de números inteiros na
linguagem C */
```

Variáveis **estáticas** são aquelas que alocam memória em tempo de *compilação*, já variáveis **dinâmicas** que alocam memória em tempo de *execução*.

Variáveis locais são aquelas que são *válidas apenas em um determinado trecho* do código, apenas dentro de uma função, por exemplo. Já **variáveis globais** são aquelas *válidas em todo o programa*. O uso de variáveis globais *não é* uma boa prática de programação.

15) Defina o que é o escopo de uma variável. Quais os escopos possíveis?

O escopo de uma variável é a faixa de sentenças nas quais ela é visível. Uma variável é visível em uma sentença se ela pode ser referenciada nessa sentença. O escopo pode ser **global** e **local**.

16) Explique o que são vetores estáticos, vetores stack-dinâmicos, vetores heap-dinâmicos de tamanho fixo e vetores heap-dinâmicos de tamanho variável.

- **Vetores estáticos**: Vinculação de memória e a definição do tamanho ocorre em tempo de compilação.
- **Vetores stack-dinâmicos**: Vinculação de memória ocorre em tempo de execução e a definição do tamanho em tempo de compilação.
- **Vetores heap-dinâmicos de tamanho fixo**: Vinculação de memória e definição do tamanho ocorre em tempo de execução (o tamanho fixo).
- **Vetores heap-dinâmicos de tamanho variável**: Vinculação de memória e definição de tamanho em tempo de execução sendo que o tamanho pode variar livremente.

17) Em algumas linguagens, os vetores são indexados a partir do 0 (zero), em outras a partir do 1. Por que é mais vantajoso indexar a partir do 0?

O cálculo do endereço para alocar o vetor é dado pela fórmula (i - inicio)*tam + EB, onde i é a posição, inicio é a primeira posição, tam é o tamanho do vetor e EB é o endereço-base. Se indexarmos a partir do zero não necessitamos realizar a operação **(i - inicio)*tam + EB** por completo, já que inicio valerá zero, assim podemos resumir a operação como **i*tam + EB**, assim, diminui-se o tempo gasto para o cálculo de endereço em 1/3.

18) O que são vetores associativos? Quais os benefícios de seu uso?

Um vetor associativo é uma coleção não ordenada de elementos de dados indexados por um número igual de valores chamado de chaves. Esta estrutura é muito melhor que um vetor caso as buscas a elementos sejam necessárias, pois a operação de dispersão implícita usada para acessar os elementos é muito eficiente. No caso dos dados serem armazenados formarem pares elas são ideais.

19) Cite 5 operações aritméticas, 6 operações relacionais e 3 operações lógicas, 3 operações de manipulação de bits, 1 operação de manipulação de strings e 2 operações de manipulação de ponteiros.

- **Operações aritméticas**: adição, subtração, multiplicação, divisão , resto da divisão.
- **Operações relacionais**: igualdade, desigualdade, maior, menor, maior ou igual, menor ou igual.
- **Operações lógicas**: conjunção, disjunção, negação.
- **Operações de manipulação de bits**: deslocamento para esquerda (SHL), deslocamento para direita (SHR), E bit-a-bit.
- **Operação de manipulação de string**: concatenação.
- **Operações de manipulação de ponteiros**: referência, derreferência.

20) Explique para que servem as operações de estreitamento e alargamento de tipo. Por que a linguagem Java não executa estreitamento implícito?

As operações de estreitamento e alargamento de tipo servem para fazer a conversão do tipo do elemento que é rvalue para o tipo do lvalue em uma operação de atribuição. Devido a Java ser uma linguagem ser fortemente tipada, ela não executa estreitamento implícito acusando erro de tipo.

21) O que é sobrecarga de operadores? Cite linguagens que permitem sobrecarga de operadores definida pelo usuário.

Sobrecarga de operadores é o uso múltiplo de um operador, ou seja, o mesmo operador é usado para fazer operações diferentes. Em C++ e C#, por exemplo, a sobrecarga de operadores está disponível ao programador.