Escola de Artes, Ciências e Humanidades (USP-Leste)
Algoritmos e Estruturas de Dados I - 2º Semestre de 2015
Prof. Alexandre da Silva Freire (`afreire@ime.usp.br` – sala 322K-I1)

Aula 20 - 23/10 - Quicksort

Correção de um exercício importante

```
void misterio1(int[] v, int p, int u) {
   if (p < u) {
      int i = fazAlgumaCoisa(v, p, u);
      misterio1(v, p, i-1);
      misterio1(v, i+1, u);
    }
}


int fazAlgumaCoisa(int[] v, int p, int u) {
   int i = p;

   for (int j = p; j < u; j++) {
      if (v[j] < v[u]) {
         troca(v, i++, j);
      }
   }

   troca(v, i, u);
   return i;
}
```

Lema 1. *O algoritmo* `fazAlgumaCoisa` *rearranja os elementos do intervalo* $v[p\ ..\ u]$ *de forma que* $v[p\ ..\ i-1] < v[i] \leq v[i+1\ ..\ u]$*, onde* i *corresponde ao índice devolvido pelo algoritmo, satisfazendo* $p \leq i \leq u$*.*

Demonstração. Note que, mesmo que a variável i fosse incrementada em todas as iterações, ainda assim, após o término do laço, teríamos que $p \leq i \leq u$. Por simplicidade, omitiremos a prova formal desse invariante.

Mostraremos que, a cada iteração do laço, imediatamente antes do teste da condição de parada, vale que $v[p\ ..\ i-1] < v[u] \leq v[i\ ..\ j-1]$.

Na primeira iteração, temos que os intervalos $v[p\ ..\ i-1]$ e $v[i\ ..\ j-1]$ são vazios e, portanto, a propriedade não é violada.

Assuma, por hipótese de indução, que a propriedade está satisfeita no início de uma iteração qualquer, exceto a última. Repare que após o término da iteração, o elemento $v[j]$ será inserido em um dos dois intervalos. Para que a propriedade seja mantida, se $v[j] < v[u]$, então $v[j]$ deveria ser inserido no intervalo que contém os números menores que $v[u]$ e, caso contrário, $v[j]$ deveria ser inserido no intervalo que contém os números maiores ou iguais a $v[u]$. No primeiro caso ($v[j] < v[u]$), após a troca de $v[i]$ com $v[j]$ e o incremento das variáveis i e j, $v[j]$ é inserido no final do intervalo $v[p \mathrel{..} i-1]$ e o intervalo $v[i \mathrel{..} j-1]$ permanece o mesmo, apenas com o elemento que estava na primeira posição sendo deslocado para a última posição. No segundo caso ($v[j] \geq v[u]$), após o incremento da variável j, o elemento $v[j]$ é inserido no final do intervalo $v[i \mathrel{..} j-1]$ e o intervalo $v[p \mathrel{..} i-1]$ permanece o mesmo. Portanto, nos dois casos, a propriedade será satisfeita no início da próxima iteração.

Após o término do laço, temos que $j = u$. Conforme o invariante demonstrado, temos que $v[p \mathrel{..} i-1] < v[u] \leq v[i \mathrel{..} j-1]$. Note que, se trocarmos $v[i]$ com $v[u]$, teremos que $v[p \mathrel{..} i-1] < v[i] \leq v[i+1 \mathrel{..} u]$. Na penúltima linha, o algoritmo troca $v[i]$ com $v[u]$ e, consequentemente, está correto. □

Lema 2. *O algoritmo* `misterio1` *ordena o intervalo* $v[p \mathrel{..} u]$.

Demonstração. Provaremos o lema por indução em $n = u - p + 1$.

Para $n = 1$, temos que o intervalo $v[p \mathrel{..} u]$ contém apenas um elemento e, portanto, já está ordenado. Nesse caso, o algoritmo não altera o intervalo e, portanto, está correto.

Para $n > 1$, assumiremos, por hipótese de indução, que o algoritmo está correto para qualquer intervalo com menos do que n elementos. Pelo Lema 1, temos que, após a chamada ao algoritmo `fazAlgumaCoisa`, os elementos do intervalo $v[p \mathrel{..} u]$ são rearranjados de forma que $v[p \mathrel{..} i-1] < v[i] \leq v[i+1 \mathrel{..} u]$, onde i corresponde ao índice devolvido pelo algoritmo, satisfazendo $p \leq i \leq u$. Note que, pela propriedade enunciada, após ordenar os intervalos $v[p \mathrel{..} i-1]$ e $v[i+1 \mathrel{..} u]$ de forma independente, teremos que o intervalo $v[p \mathrel{..} u]$ estará ordenado. Note que, como $p \leq i \leq u$, temos que $(i-1)-p+1 = i-p < u-p+1 = n$ e $u-(i+1)+1 = u-i < u-p+1 = n$. Consequentemente, por hipótese de indução, as chamadas recursivas ordenam os intervalos $v[p \mathrel{..} i-1]$ e $v[i+1 \mathrel{..} u]$. Portanto, o algoritmo está correto. □

A seguir, apresentamos novamente as mesmas provas, porém com algumas dicas adicionais sobre como construir provas matemática.

Lema 3. **(ETAPA A - enunciado do problema)** { *O algoritmo* `fazAlgumaCoisa` *rearranja os elementos do intervalo* $v[p \mathrel{..} u]$ *de forma que* $v[p \mathrel{..} i-1] < v[i] \leq v[i+1 \mathrel{..} u]$, *onde* i *corresponde ao índice devolvido pelo algoritmo, satisfazendo* $p \leq i \leq u$. }

Demonstração. **(propriedades que são facilmente verificáveis não necessitam de demonstração) {** Note que, mesmo que a variável i fosse incrementada em todas as iterações, ainda assim, após o término do laço, teríamos que $p \leq i \leq u$. Por simplicidade, omitiremos a prova formal desse invariante. }

(ETAPA B - enunciados dos invariantes) { Mostraremos que, a cada iteração do laço, imediatamente antes do teste da condição de parada, vale que $v[p \,..\, i-1] < v[u] \leq v[i \,..\, j-1]$. }

(ETAPA C - prova dos invariantes por indução) {

(ETAPA C1 - inicialização – caso base) Na primeira iteração, temos que os intervalos $v[p \,..\, i-1]$ e $v[i \,..\, j-1]$ são vazios e, portanto, a propriedade não é violada.

(ETAPA C2 - manutenção – passo da indução) Assuma, por hipótese de indução, que a propriedade está satisfeita no início de uma iteração qualquer, exceto a última . Repare que após o término da iteração, o elemento $v[j]$ será inserido em um dos dois intervalos.

(fato que independe do que o algoritmo faz) Para que a propriedade seja mantida, se $v[j] < v[u]$, então $v[j]$ deveria ser inserido no intervalo que contém os números menores que $v[u]$ e, caso contrário, $v[j]$ deveria ser inserido no intervalo que contém os números maiores ou iguais a $v[u]$.

(argumento – compara o que o algoritmo faz com o fato enunciado anteriormente) No primeiro caso ($v[j] < v[u]$), após a troca de $v[i]$ com $v[j]$ e o incremento das variáveis i e j, $v[j]$ é inserido no final do intervalo $v[p \,..\, i-1]$ e o intervalo $v[i \,..\, j-1]$ permanece o mesmo, apenas com o elemento que estava na primeira posição sendo deslocado para a última posição. No segundo caso ($v[j] \geq v[u]$), após o incremento da variável j, o elemento $v[j]$ é inserido no final do intervalo $v[i \,..\, j-1]$ e o intervalo $v[p \,..\, i-1]$ permanece o mesmo. Portanto, nos dois casos, a propriedade será satisfeita no início da próxima iteração. **(invariante está provado)**
}

(ETAPA D - término = invariante + condição de parada ⇒ corretude) Após o término do laço, temos que $j = u$.

(fato que independe do que o algoritmo faz) Conforme o invariante demonstrado, temos que $v[p \,..\, i-1] < v[u] \leq v[i \,..\, j-1]$. Note que, se trocarmos $v[i]$ com $v[u]$, teremos que $v[p \,..\, i-1] < v[i] \leq v[i+1 \,..\, u]$.

(argumento – compara o que o algoritmo faz com o fato enunciado anteriormente) Na penúltima linha, o algoritmo troca $v[i]$ com $v[u]$ e, consequentemente, está correto. □

Lema 4. *O algoritmo* `misterio1` *ordena o intervalo* $v[p \,..\, u]$.

Demonstração. **(diga em qual parâmetro você fará a indução)** Provaremos o lema por indução em $n = u - p + 1$.

(escolha como caso base o menor caso que o seu algoritmo deve tratar - verifique se apenas um caso base é suficiente!) Para $n = 1$, temos que o intervalo $v[p \mathrel{..} u]$ contém apenas um elemento e, portanto, já está ordenado. Nesse caso, o algoritmo não altera o intervalo e, portanto, está correto.

(enuncie qual é a hipótese de indução) Para $n > 1$, assumiremos, por hipótese de indução, que o algoritmo está correto para qualquer intervalo com menos do que n elementos.

(se precisar utilizar algum outro lema auxiliar, enuncie a propriedade que será utilizada nesta prova) Pelo Lema 1, temos que, após a chamada ao algoritmo `fazAlgumaCoisa`, os elementos do intervalo $v[p \mathrel{..} u]$ são rearranjados de forma que $v[p \mathrel{..} i-1] < v[i] \leq v[i+1 \mathrel{..} u]$, onde i corresponde ao índice devolvido pelo algoritmo, satisfazendo $p \leq i \leq u$.

(fato que independe do que o algoritmo faz) Note que, pela propriedade enunciada, após ordenar os intervalos $v[p \mathrel{..} i-1]$ e $v[i+1 \mathrel{..} u]$ de forma independente, teremos que o intervalo $v[p \mathrel{..} u]$ estará ordenado.

(para utilizar a hipótese de indução, o parâmetro da chamada recursiva precisa ser menor do que n) Note que, como $p \leq i \leq u$, temos que $(i-1)-p+1 = i-p < u-p+1 = n$ e $u-(i+1)+1 = u-i < u-p+1 = n$.

(argumento – compara o que o algoritmo faz com o fato enunciado anteriormente) Consequentemente, por hipótese de indução, as chamadas recursivas ordenam os intervalos $v[p \mathrel{..} i-1]$ e $v[i+1 \mathrel{..} u]$. Portanto, o algoritmo está correto. $\square$