Imperative Programming II (Sheet #2)

Marius Gavrilescu

1. Works for "b". Does not work for "ca". Simplest change is to use two buffers.

2.
```
var i = 0
while (i < t.length) {
  if (t.charAt(i) == 'a') {
    t.insert(i+1, 'b')
    i = i + 1
  } else if (t.charAt(i) == 'c') {
    t.deleteChar(i)
    i = i - 1
  }
  i = i+1
}
```

3. $2N$ where N is the number of characters read.

4. Not defined for $len = MAX + 1$, not one-to-one (abs("a...b", 2, 1) = abs("ab...", 2, 2)), and its image is all strings of length at most MAX.

5.
```
class BoolIntSet {
  private val MAX = 1000;
  private val freq = new Array[Boolean](MAX);
  private var size = 0;
  // abstraction function: members = { i | 0 <= i < MAX && freq(i) }

  def isEmpty = freq.exists(_)

  def contains(x : Int) = {
    require(x >= 0 && x < MAX)
    freq(x)
  }

  def insert(x : Int) = {
    require(x >= 0 && x < MAX)
    freq(x) = true
  }

  def delete(x : Int) = {
    require(x >= 0 && x < MAX)
    freq(x) = false
  }
}
```

New precondition `PRE 0 <= x < MAX` on contains, insert, delete.

6.
```
STATE text : Seq[Char]

clear()
POST text = []

isEmpty = r : Boolean
POST r = (text == [])

put(ch : Char)
POST text = text0 ++ [ch]

get() = r : Char
PRE !isEmpty
POST r = head text
POST text = tail text0
```

7.
```
var ar = new Array[Char](100);
var size = 0;
// abstraction function: text = ar[0..size)

def clear() = (size = 0)
def isEmpty = (size == 0)

def put(ch : Char) = {
  assume(size < 100);
  ar(size) = ch;
  size = size + 1;
}

def get() = {
  require(!isEmpty);
  val r = ar(0);
  size = size - 1;
  for(i <- 0 until size)
    ar(i) = ar(i + 1);
  r
}
```