```
    1: class Numeric
    2:   def consonant?
    3:     (0x3131..0x314e).include?(self)
    4:   end
    5:
    6:   def vowel?
    7:     (0x314f..0x3163).include?(self)
    8:   end
    9: end
   10:
   11: class Phoneme
   12:   def initialize
   13:     @consonant ||= {
   14:       # compat    cho     jong
   15:       0x3131 => [0x1100, 0x11A8],  # ã\204±
   16:       0x3132 => [0x1101, 0x11A9],  # ã\204²
   17:       0x3133 => [0x0000, 0x11AA],  # ã\204±ã\205\205
   18:       0x3134 => [0x1102, 0x11AB],  # ã\204´
   19:       0x3135 => [0x0000, 0x11AC],  # ã\204´ã\205\210
   20:       0x3136 => [0x0000, 0x11AD],  # ã\204´ã\205\216
   21:       0x3137 => [0x1103, 0x11AE],  # ã\204•
   22:       0x3138 => [0x1104, 0x1104],  # ã\204¸
   23:       0x3139 => [0x1105, 0x11AF],  # ã\204¹
   24:       0x313A => [0x11B0, 0x11B0],  # ã\204¹
   25:       0x313B => [0x0000, 0x11B1],  # ã\204¹ã\205\201
   26:       0x313C => [0x0000, 0x11B2],  # ã\204¹ã\205\202
   27:       0x3141 => [0x1106, 0x11B7],  # ã\205\201
   28:       0x3142 => [0x1107, 0x11B8],  # ã\205\202
   29:       0x3143 => [0x1108, 0x0000],  # ã\205\203
   30:       0x3145 => [0x1109, 0x11BA],  # ã\205\205
   31:       0x3146 => [0x110A, 0x11BB],  # ã\205\206
   32:       0x3147 => [0x110B, 0x11BC],  # ã\205\207
   33:       0x3148 => [0x110C, 0x11BD],  # ã\205\210
   34:       0x3149 => [0x110D, 0x0000],  # ã\205\211
   35:       0x314A => [0x110E, 0x11BE],  # ã\205\212
   36:       0x314B => [0x110F, 0x11BF],  # ã\205\213
   37:       0x314C => [0x1110, 0x11C0],  # ã\205\214
   38:       0x314D => [0x1111, 0x11C1],  # ã\205\215
   39:       0x314E => [0x1112, 0x11C2]   # ã\205\216
   40:     }
   41:   end
   42:
   43:   # 3ê°\234 ì\227°ì\206\215 ì\236\220ì\235\214ì\235´ ë\202\230ì\230¤ë\212\224 ê²½ì
\232°
   44:   def compact_cons(codes)
   45:     i, result = 0, []
   46:
   47:     while i < codes.length - 2
   48:       first, second, third = codes[i], codes[i+1], codes[i+2]
   49:
   50:       incr = 1
   51:       if [first, second, third].all? { |e| e.consonant? }
   52:         if first == 0x3131                    # ã\204±
   53:           case second
   54:           when 0x3145 then result.push(0x3133); incr = 2 # ã\204³
   55:           else             result.push(first)
   56:           end
   57:         elsif first == 0x3134                 # ã\204´
   58:           case second
   59:           when 0x3148 then result.push(0x3135); incr = 2 # ã\204µ
   60:           when 0x314E then result.push(0x3136); incr = 2 # ã\204µ
   61:           else             result.push(first)
   62:           end
   63:         elsif first == 0x3139                 # ã\204¹
   64:           case second
   65:           when 0x3131 then result.push(0x3162); incr = 2 # ã\204º
   66:           when 0x3141 then result.push(0x313B); incr = 2 # ã\204»
   67:           when 0x3142 then result.push(0x313C); incr = 2 # ã\204¼
```

```
 68:           when 0x3145 then result.push(0x313D); incr = 2 # ã\204½
 69:           when 0x314C then result.push(0x313E); incr = 2 # ã\204¾
 70:           when 0x314D then result.push(0x313F); incr = 2 # ã\204¿
 71:           else             result.push(first)
 72:           end
 73:         else
 74:           result.push(first)
 75:         end
 76:       else
 77:         result.push(first)
 78:       end
 79:
 80:       i += incr
 81:     end
 82:
 83:     return result + codes.last(2)
 84:   end
 85:
 86:   def compact_vowels(codes)
 87:     i, incr, result = 0, 0, []
 88:
 89:     while i < codes.length - 1
 90:       first, second = codes[i], codes[i+1]
 91:       incr = 1
 92:       if first == 0x3157                    # ã\205\227
 93:         case second
 94:         when 0x314F then result.push(0x3158); incr = 2 # ã\205\230
 95:         when 0x3150 then result.push(0x3159); incr = 2 # ã\205\231
 96:         when 0x3163 then result.push(0x315A); incr = 2 # ã\205\232
 97:         else             result.push(first)
 98:         end
 99:       elsif first == 0x315C                 # ã\205\234
100:         case second
101:         when 0x3153 then result.push(0x315D); incr = 2 # ã\205\235
102:         when 0x3154 then result.push(0x315E); incr = 2 # ã\205\236
103:         when 0x3163 then result.push(0x315F); incr = 2 # ã\205\237
104:         else             result.push(first)
105:         end
106:       elsif first == 0x3161                 # ã\205¡
107:         case second
108:         when 0x3163 then result.push(0x3162); incr = 2 # ã\205¢
109:         else             result.push(first)
110:         end
111:       else
112:         result.push(first)
113:       end
114:
115:       i += incr
116:     end
117:
118:     result.push(codes.last) if incr == 1
119:
120:     return result
121:   end
122:
123:   def to_jamo(compat_jamo)
124:     res = compat_jamo.each_cons(2).map do |e|
125:       pos = (e[0].consonant? and e[1].consonant?) ? 1 : 0
126:       translate(e[0], pos)
127:     end
128:
129:     last = compat_jamo.last
130:     pos  = last.consonant? ? 1 : 0
131:     res.push(translate(last, pos))
132:
133:     return res
134:   end
135:
```

```
136:   def translate(code, pos=0)
137:     return 0 if (0x3131..0x314e).include?(code) and pos > 1
138:
139:     case code
140:     when 0x3131..0x314e then @consonant[code][pos]
141:     when 0x314f..0x3163 then code - 8174
142:     else
143:       0
144:     end
145:   end
146: end
147:
148: =begin rdoc
149: @vowel ||= {
150:   # compat
151:   0x314F => 0x1161,  # ã\205\217
152:   0x3150 => 0x1162,  # ã\205\220
153:   0x3151 => 0x1163,  # ã\205\221
154:   0x3152 => 0x1164,  # ã\205\222
155:   0x3153 => 0x1165,  # ã\205\223
156:   0x3154 => 0x1166,  # ã\205\224
157:   0x3155 => 0x1167,  # ã\205\225
158:   0x3156 => 0x1168,  # ã\205\226
159:   0x3157 => 0x1169,  # ã\205\227
160:   0x3158 => 0x116A,  # ã\205\230
161:   0x3159 => 0x116B,  # ã\205\231
162:   0x315A => 0x116C,  # ã\205\232
163:   0x315B => 0x116D,  # ã\205\233
164:   0x315C => 0x116E,  # ã\205\234
165:   0x315D => 0x116F,  # ã\205\235
166:   0x315E => 0x1170,  # ã\205\236
167:   0x315F => 0x1171,  # ã\205\237
168:   0x3160 => 0x1172,  # ã\205
169:   0x3161 => 0x1173,  # ã\205¡
170:   0x3162 => 0x1174,  # ã\205¢
171:   0x3163 => 0x1175   # ã\205£
172: }
173: =end
```