Algorithm 1: runBRKGA

```
KeysSorting = no intervalo [0,1] * TAMPOP
Population = decoder(KeysSorting)
while $(g \leq QTDGE)$ do
    Population.sort()
    PopulationG = Population[0...TAMELI]
    for $i \leftarrow TAMMUT$ do
        (parentBest, parentRandom) = getParents(KeysSorting)
        child = crossover(parentBest, parentRandom)
        PopulationG = child
    end
    for $i \leftarrow (TAMPOP - (TAMELI + TAMMUT))$ do
        (parentBest, parentRandom) = getParents(KeysSorting)
        child = crossover(parentBest, parentRandom)
        PopulationG = child
    end
    PopulationG = decoder(KeysSorting)
    Population ← PopulationG
    g = g+1
    end
```

Algorithm 2: decoder

```
for $m \leftarrow TAMPOP$ do
    collided = False
    usedColors = [0]
    KeysSorting[m].sort()
    for $c \leftarrow KeysSorting[m]$ do
        index = KeysSorting[m].index(c)
        color = usedColors[len(usedColors)-1]
        if graph.vertex[index].color == incolor then
            graph.vertex[index].color = color
        end
        colorNeighbor = getcolorNeighbor(c)
        if graph.vertex[index].color in colorNeighbor then
            *Tenta reutilizar uma das cores da paleta de cores.
            Se não der, cria uma nova cor.
            Atribui a cor selecionada.*
            graph.vertex[index].color = color
        end
        for $i \leftarrow graph.AmoutVertex$ do
            *Para todos não adjacentes ao vertice atual, recebe a mesma
              cor*
        end
    end
end
```