func triangleFree2Coloring($G = (V, E)$)

// Dieser Algorithmus berechnet eine dreiecksfreie 2-Färbung von des Graphen $G = (V, E)$

Färbe G *zufällig* mit zwei Farben;

while G enthält ein monochromes Dreieck T **do**

    Färbe einen zufälligen Knoten aus T um;

od

// Die aktuelle 2-Färbung von G *ist dreiecksfrei*

<u>**proc** simulatedAnnealing</u>

Beginne bei einer zufälligen Lösung s;

for eine gewisse Anzahl von Schritten **do**

Wähle (gleichverteilt) einen zufälligen Nachbarn

$s' \in N(s)$;

$d := val(s') - val(s)$;

if $d > 0$ **then**

$s := s'$;

else

r := gleichverteilte Zufallszahl auf $[0, 1]$;

// im folgenden Term ist T eine geeignete

// Konstante

if $r < e^{d/T}$ **then**

$s := s'$;

fi

fi

od

// Das beste bislang gefundene s ist eine gute

// Approximation für optimale Lösung s_{max}