Algorithms 2021/2022

June Exam

Solve the following exercises.

1. Consider the following statement: "*Let $G(V, E, w)$ be a connected undirected graph with a* distinct *positive* weight *$w(e)$ on each edge $e \in E$. Let $e^* \in E$ be the edge of the graph of highest weight. Then, no minimum spanning tree of the graph contains e^*.*"

   Determine whether the statement is true or false: if it is true, prove it; if it is false, give a counterexample.

2. In a country there are three types of coin: the first one has value a, the second has value b and the third has value c. (Here, a, b, c are positive integers). In the country, a price w (again, a non-negative integer) can be paid exactly if and only if there exist three non-negative integers i_a, i_b, i_c such that $w = i_a \cdot a + i_b \cdot b + i_c \cdot c$ — that is, such that w equals the total value of i_a coins of value a, i_b coins of value b and i_c coins of value c. Write an algorithm that, given a, b, c and a price w, returns `True` if w can be paid exactly, and `False` otherwise. *(More points will be awarded to faster solutions.)*

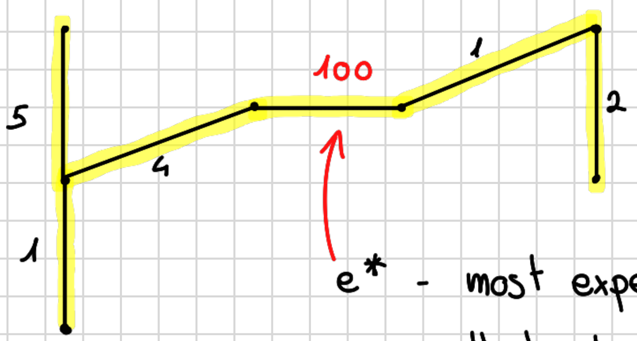
   Example 1: if $a = 1, b = 5, c = 7$ and $w = 13$, then the function should return `True` — indeed, to pay w, one could use 13 coins of value a, given that $w = 13 \cdot a$. Or, one could use 1 coin of value c, 1 coin of value b, and 1 coin of value a, given that $w = 1 \cdot a + 1 \cdot b + 1 \cdot c$.

   Example 2: if $a = 2, b = 4, c = 8$ and $w = 7$, then the function should return `False` (there are no three non-negative integers i_a, i_b, i_c such that $w = i_a \cdot a + i_b \cdot b + i_c \cdot c$).

CamScanner

ex. 1

FALSE. Let's see a <u>counterexample</u>:



= MST

e* - most expensive edge that should necessarily be part of the MST, otherwise, if e* wasn't considered, the graph wouldn't be connected anymore.

ex. 2

payThePrice(a, b, c, w):

Iterative basic solution

```
if w == 0
    return FALSE
elif
    a, b, c == 1   OR   a,b,c == w
        return TRUE
else
    for i_a 0, ..., w
        for i_b 0, ..., w
            for i_c 0, ..., w

                if i_a · a + i_b · b + i_c · c == w
                    return TRUE
            else
                return FALSE
```

$O(w^3)$

```
payThePrice(a,b,c,w):

    if w == 0                        Iterative solution
        return FALSE                 having a faster runtime
    elif
        a, b, c == 1   OR   a,b,c == w
            return TRUE
    else
O(w²)   for ia 0, ..., w
            for ib 0, ..., w
                x = w - (ia·a + ib·b)
                if x % c == 0
                    return TRUE
                else
                    return FALSE
```

```
payThePrice(a,b,c,w):

    x = [FALSE] * (w + 1)            Dynamic p.
    x[0] = TRUE                      solution

O(w)  for i 1, ..., w
          if (i >= a AND x[i - a]) OR
             (i >= b AND x[i - b]) OR
             (i >= c AND x[i - c]):
              x[i] = TRUE
      return x[w]
```