Aluna: Caroline Viana Bento
ADO 4 - Teoria da Computação - BCC 6° Semestre

1. A linguagem SOMA – SUBC é definida como segue:

$SOMA - SUBC = \{ < S, t > | S = \{x1,...,xk\}$ *e para algum* $\{y1, \ldots, yl\} \subseteq \{x1, \ldots, xk\}$, *temos* $\sum y_i = t \ \}$

Escreva um programa que dado um conjunto S *e um valor* t *lido como entrada, determine se* $< S, t > \in SOMA - SUBC$ *. Caso positivo, apresente o subconjunto encontrado. Caso contrário apresente uma mensagem indicando que não há subconjunto que atenda os requisitos.*
Exemplo:
entrada: $S = \{1,3,6,9\}, t = 13$
saída: $\{1,3,9\}$

entrada:$S = \{1,5,7,9\}, t = 4$
saída: Não existe subconjunto de S *cujas somas dos elementos resulta em 4.*

Prova:

Link Replit: https://replit.com/@CarolineViana/SomaSubc#main.py
Link Github: https://github.com/carolinevianab/SomaSubc

2. Mostre que P é fechado sobre concatenação.

Prova:
Sejam L_1 e $L_2 \ \in P$. Seja M_1 e M_2 MTs que decidem L_1 e L_2, respectivamente, em tempo polinomial. Construimos uma M_3 que decida a concatenação de L_1 com L_2 em tempo polinomial.

Seja w a entrada tal que $w = a_1 + a_2 + a_3 + \ldots + a_n$, seja i um valor incremental tal que $0 < i < n$, e seja a_i um caractere de w.

M_3 = "Sobre a entrada w":
- Para cada valor de i, seja $w_1 = a_1 + a_2 + a_3 + \ldots + a_i$ e $w_2 = a_i + a_{i+1} + a_{i+2} + \ldots + a_n$, tal que $w_1 \in L_1$ e $w_2 \in L_2$
- Execute M_1 sobre w_1. Se aceita, prossiga para a próxima etapa. Se rejeita, rejeite.
- Execute M_2 sobre w_2. Se aceita, aceite. Caso contrário, rejeite.

M_3 aceita w se e somente se M_1 aceita w_1 e M_2 aceita w_2. Logo, M_3 aceita a concatenação de L_1 e L_2. Como M_1 e M_2 executam em tempo polinomial, e a etapa 1 executará no máximo $n + 1$ vezes - ou seja, tempo $O(n)$ - M_3 também executa em tempo polinomial.

c.q.d.

3. Mostre que NP é fechado sobre concatenação.

Prova:
Sejam L_1 e $L_2 \in P$. Seja M_1 e M_2 MTNs que decidem L_1 e L_2, respectivamente, em tempo polinomial. Construimos uma M_3 que decida a concatenação de L_1 com L_2 em tempo polinomial.

Seja w a entrada tal que $w = a_1 + a_2 + a_3 + \ldots + a_n$, seja i um valor incremental tal que $0 < i < n$, e seja a_i um caractere de w.

M_3 = "Sobre a entrada w":
- Para cada valor de i, seja $w_1 = a_1 + a_2 + a_3 + \ldots + a_i$ e $w_2 = a_i + a_{i+1} + a_{i+2} + \ldots + a_n$, tal que $w_1 \in L_1$ e $w_2 \in L_2$
- Execute M_1 sobre w_1. Se aceita, prossiga para a próxima etapa. Se rejeita, rejeite.
- Execute M_2 sobre w_2. Se aceita, aceite. Caso contrário, rejeite.

M_3 aceita w se e somente se M_1 aceita w_1 e M_2 aceita w_2. Logo, M_3 aceita a concatenação de L_1 e L_2. Como M_1 e M_2 executam em tempo polinomial, e a etapa 1 executará no máximo $n + 1$ vezes - ou seja, tempo $O(n)$ - M_3 também executa em tempo polinomial.

c.q.d.