Konfigurace

Řešení problému

Potřebujeme zjistit jaké řetězce spustí všechny skripty.

Hledání řetězců

Co budeme využívat?

Z inputu i-tých bitů, které ovlivňují skripty si uděláme slovník, který bude mýt i jako klíč a list ovlivněných skriptů jako hodnotu.

Jak zjistíme validní řetězce?

Budeme generovat řetězce přes „**Generování řetězce**“ a vždy použijeme „**Ověření řetězce**“, po ukončení vypíšeme délku listu získaného z „**Ověření řetězce**“ a první řetězec z tohoto listu.

Generování řetězce

Vygenerujeme si všechny možné kombinace řetězců a u všech zavoláme funkci „Ověření řetězce“

Ověření řetězce

Použijeme uložený slovník bitů a vypočítáme kolikrát řetězec ovlivňuje jaký skript. Poté projdeme všechny tyto sumy a zkontrolujeme, zda je tam aspoň jedna sudá a pokud ano tak tento řetězec je vadný (má aspoň jeden skript, který není zapnutý), pokud vadný není, uložíme ho.

Zdůvodnění řešení

Toto řešení bude vždy funkční, protože kontrolujeme úplně všechny kombinace řetězců, které jdou poslat.

Zdůvodnění nejlepšího řešení

Toto řešení je nejlepší, protože si můžeme být vždy jistí, že bude 100% správně, díky tomu že prochází všechny řešení.

Časová komplexita

Ověřování validního řetězce má časovou komplexitu O(n). Procházení řetězců má komplexitu O(2^m).

Tyto funkce jsou vykonány v sobě, ověřování je provedeno pro každý řetězec, takže celková časová komplexita tohoto algoritmu je tedy O(n*2^m).