Oscar Azrak

oazrak@kth.se

DD1320 Tillämpad datalogi - KTH HT 2020

## 1 Uppgiftsbeskrivning

Matchsticks are ideal tools to represent numbers. A common way to represent the ten decimal digits with matchsticks is the following:



This is identical to how numbers are displayed on an ordinary alarm clock. With a given number of matchsticks you can generate a wide range of numbers. We are wondering what the smallest and largest numbers are that can be created by using all your matchsticks.

## Input

On the first line one positive number: the number of testcases, at most 100. After that per testcase:

* One line with an integer 𝑛 (2 ≤ 𝑛 ≤ 100): the number of matchsticks you have.

## Output

Per testcase:

* One line with the smallest and largest numbers you can create, separated by a single space. Both numbers should be positive and contain no leading zeroes.

## 2 Kattis

https://kth.kattis.com/submissions/6507271

## 3 Datastrukturer

De datastrukturer som använts i detta program är listor och integers (heltal).

## 4 Algoritm

Indata: Antal tändstickor

Utdata: Lägsta talet och högsta talet för antalet stickor

1. Spara antalet tändstickor ett tal kräver i en lista från noll till nio, så att indexet i listan motsvarar siffran och elementet motsvarar antalet stickor.
2. Bestäm hur många siffror N som måste användas för totala antalet tändstickor.
3. Skapa en variabel som sparar hur många tändstickor som är kvar genom att först sätta den lika med totala mängden stickor.
4. Skapa en variabel resultat som är lika med noll.
5. Upprepa punkt 6-8 för varje siffra, totalt N gånger.
6. Ta fram det minsta antalet stickor som går att använda för respektive siffra.
7. Ta fram max antalet tändstickor som går att använda för respektive siffra.
8. Inga ledande nollor så två fall måste undersökas, ett där resultat redan är noll och ett fall då resultatet ej är noll.
   1. Hitta antalet stickor som är lämpligt i intervallet

[minsta antalet stickor, max antalet stickor] genom att gå igenom listan med antalet tändstickor varje tal behöver.

* 1. När antalet stickor hittats, addera dess index till resultatet och subtrahera antalet tändstickor som är kvar med hur många tändstickor som användes för det talet (elementet för det indexet).

## 5 Tidskomplexitet

I min kod har jag en två funktioner med två stycken for-loops i varandra. Loopar från 0 till n medan den andra loopar från 1 till 10 eller 0 till 9. Detta gör att tidskomplexiteten blir O(10\*n) där O, ”Big O Notation” används för att klassificera algoritmer enligt hur snabba de är eller hur mycket utrymme som krävs och n är storleken. I det här fallet används O-noteringen för att avgöra tidskomplexiteten. I ”main” funktionen finns det dessutom en till for-loop som loopar från 0 till n, beroende på hur många inputs man ska ta emot. Denna funktion har tidskomplexitet O(n), detta gör att man kan addera O(10\*n) med O(n) som blir O(11\*n) men detta kan förenklas till O(n) eftersom n kan bli hur stort som helst så spelar konstanterna ingen roll.

Tidskomplexiteten för min kod är O(n)

## Appendix1: ProgramkodEn bild som visar text Automatiskt genererad beskrivningEn bild som visar text Automatiskt genererad beskrivningEn bild som visar text Automatiskt genererad beskrivning

## Appendix2: Testdata

Den första testdata som testades är om första input är noll eller mindre ska programmet be om en ny input tills en godkänd input ges då ska programmet ta emot input för antalet stickor.

Den andra testdatan är om man anger noll stickor eller en sticka, då ska programmet be om ny input tills användaren ger korrekt input.

Och den tredje testdatan är att testa om programmet tillåter ledande nollor, så först testas 6 stickor och sedan 14 och 8, om den skulle tillåta med ledande nollor är den lägsta siffran 001 respektive 01.

I alla tre fall får endast inputs som kan omvandlas till integers tillåtas, inga bokstäver, specialtal eller speciella tecken.

Så testdatan är:

0 🡨 testar om antal rader som ska läsas in efter är 1 eller mer, om de är mindre provas en input

-1 🡨 testar som ovan, om siffran man ger i input är mindre än 1 ska den försöka igen

+ 🡨 programmet ska endast tillåta siffror

tilda 🡨 testar som ovan, programmet ska endast tillåta siffror

4 🡨 en tillåten input

1 🡨 antal tändstickor måste vara två till hundra, så under 2 tillåts ej

2.5 🡨 Tillåter ej decimaltal

2 🡨 tillåten input

1000 🡨 som ovan, input måste vara mellan 2 och 100

101

0 🡨 testar att programmet ej tillåter fel input trots en tillåten input angetts tidigare

6 🡨 får ej ge 0 som sv1ar, måste ge 6

14 🡨 testar om den ger 001 eller 88 som den bör

8 🡨 testar om den ger 01 eller 10 som den bör

En bild som visar objekt, klocka

Automatiskt genererad beskrivning

Programmet ignorerar de felaktiga inputsen och tar endast in de korrekta inputs och ger do ut korrekta outputs