Algoritmi i strukuture podataka

Algoritam je sekvenca definisanih računarskih procedure koja prima ulaz i daje izlaz. Ulaz pretvara u izlaz. Opisuje specifičnu računarsku proceduru za dobijanje rezultata izlaza.

Ulaz: Sekvenca n brojeva

Izlaz: Permutacija tih brojeva, tkd a1<a2...<an

Ovakva sekvenca ulaza se naziva instanca problema sortiranja.   
Instanca problema se sastoji od ulaza koji zadovoljava neka ograničenja, koji je potreban da bi se izračunalo rešenje problema.

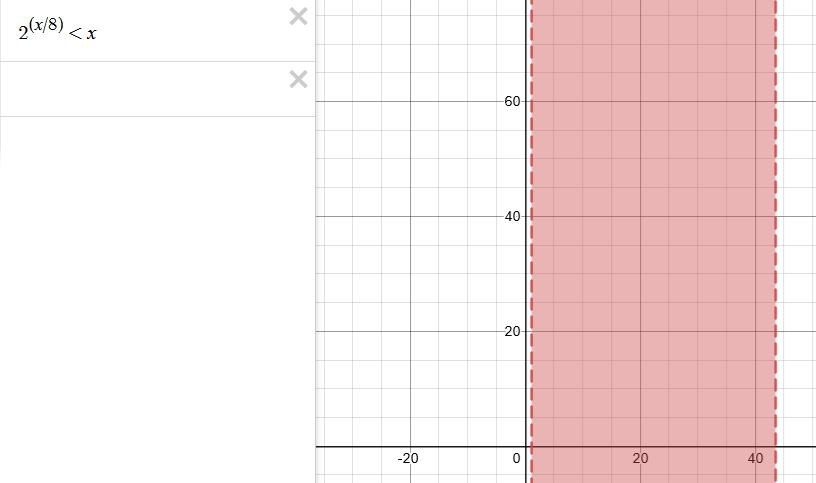
Algoritam je tačan ako za svaku instancu ulaza daje tačan izlaz. Algoritam ne mora biti samo softverski, može biti izdizajniran hardverski.

Nemaju svi problemi rešenje. Ovakvi problemi se zovu NP-kompletni.

**1.2-2:**

8n^2 < 64n\*lg(n)

n < 8lg(n)

2^(n/8) < n 

**1.2-3**:

2^2\*5^2 < 2^x

(2\*5x)^2 < 2^x

10x < 2^x/2

X < 2^(x/2) / 10 => 10 < 2^(x/2)/x…… x/2 = t 10<sqr(2) x/2=sqr(2)

**2.1-4:**

Array A, B cuvaju integer u binarnoj formi

Izracunaj sumu A I B I rezultat smesti u C

// Korisnik upisuje 2 broja sa standardnog input stream-a

Function decimal2Binary(a int) []int {

Bits = new array[]int {}

While a > 0 {

Mod = a % 2

a = a / 2

if mod > 0 {

bits.append(1)

} else {

Bits.append(0)

}

}

Return Bits;

}

Function binarySum(a, b int[]) int[] {

Newarr = []int{}

minLen = min(len(a), len(b)

left = 0

for I = 0; I < minLen; i++ {

sum = a[i] + b[i] + left

if sum == 2 {

left = 1

Newarr.append(0)

} else {

Left = 0

NewArr.append(1)

}

}

Longer = len(a) > len(b) ? a : b

Return newarr.append(longer[len(newarr):])

}

**2.2-1:** n^3/1000-100n^2-100n+3 = Theta(n^3)

**2.2-2:**   
 a) srednja vremenska kompleksnost: N/2  
 b) najgora vremenska kompleksnost: N  
 c) Obe Theta notacije su Theta(N)

**2.2-4:** Metodom early returna, ili da njegov najbolji slucaj vremenske kompleksnosti bude specijalan slucaj, gde se vraca resenje odmah.

**Divide and conquer algoritmi (Povedi i vladaj :D ):**

* **Podeli** problem u vise manjih problema, koji su manje instance istog problema
* **Vladaj** problemima resavajuci ih rekurzivno. Ako su problem dovoljno mali, resi ih na najprostiji nacin, mozda i bez rekurzije.
* **Kombinuj** resenja vladanja u resenje originalnog problema

**Merge sort (Theta(N)):**

* **Podeli** sekvencu od N elemenata u N/2
* **Vladaj –** sortiraj podsekvence rekurzivno
* **Kombinuj** resenja i napravi odgovor
* **Zavrsi** kada sekvenca dostigne duzinu 1

**2.3-3: Dokazi da vazi sledece**

N = 2 =>