Example

```
int search(int a[],int &n, int x)
{    //a[] is an array of n elements
  for (int i=0; i<n; i++)
         if (a[i] == x) return i;
  return -1;
}
```

- Run-time behaviour:
  - Best case
  - average case
  - **worst case**

Asymptotic notation

- $T_A(n) = c_1 . n^2 + c_2 . n$
- $T_B(n) = c_3 . n$
- For large n, Alg B is superior to Alg A, regardless of the value of c_1, c_2, c_3.
- I.e.

$$\exists\, \mathbf{n_o} : \forall\, \mathbf{n} \geq \mathbf{n_o} : \mathbf{T_A(n)} \geq \mathbf{T_B(n)}$$

Asymptotic notation

- $T_C(n) = 4 . n^2 + 10 . n$
- $T_D(n) = 6 . n^2$
- For large n, Alg C is superior to Alg D
- I.e.

$$\exists\, \mathbf{n_o} : \forall\, \mathbf{n} \geq \mathbf{n_o} : \mathbf{T_D(n)} \geq \mathbf{T_C(n)}$$

- But...

$$\forall\, \mathbf{n} : \mathbf{T_D(n)} \leq \mathbf{[6/4]} \,.\, \mathbf{T_C(n)}$$

- i.e., even for large n, Alg D is "worse" by *at most*

Asymptotic notation

$T_A(n) = c_1 . n^2 + c_2 . n$

$T_B(n) = c_3 . n$

$T_C(n) = 4. n^2 + 10 . n$

$T_D(n) = 6 . n^2$

- We consider Algorithm B to be *better than* Algorithms A, C and D.
- Formalized in "Big Oh" notation: