Example

```c
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

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

\[ \exists n_0: \forall n \geq n_0 : T_A(n) \geq T_B(n) \]
Asymptotic notation

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

$$\exists n_0: \forall n \geq n_0: T_D(n) \geq T_C(n)$$

• But...

$$\forall n: T_D(n) \leq [6/4] \cdot T_C(n)$$

• i.e., even for large $n$, Alg D is “worse” by at most
Asymptotic notation

\[ T_A(n) = c_1 \cdot n^2 + c_2 \cdot n \]
\[ T_B(n) = c_3 \cdot n \]
\[ T_C(n) = 4 \cdot n^2 + 10 \cdot n \]
\[ T_D(n) = 6 \cdot n^2 \]

- We consider Algorithm B to be *better than* Algorithms A, C and D.

- Formalized in ““Big Oh”” notation: