Sortowanie szybkie (Quicksort)

```
Partition(A,p,r)
  x=A[r] //element wyznaczajacy podział
  i=p-1
  for j=p to r
    if A[j]<=x
       i=i+1
       zamien A[i] z A[j]
  if i<r return i
  else   return i-1


Quicksort(A,p,r)
  if p<r
     q=Partition(A,p,r)
     Quicksort(A,p,q)
     Quicksort(A,q+1,r)
```

Stwierdzenie Pesymistyczny czas sortowania szybkiego jest $\Theta(n^2)$, gdzie n to rozmiar sortowanej tablicy.

RandomizedQuickSort

Sortowanie szybkie z losowym wyborem elementu wyznaczającego podział

```
RandomizedPartition(A,p,r)
  wylosuj wartość k z zakresu p,p+1,...,r
  zamień A[k] z A[r]
  return Partition(A,p,r)
```

```
RandomizedQuicksort(A,p,r)
  if p<r
    q=RandomizedPartition(A,p,r)
    RandomizedQuicksort(A,p,q)
    RandomizedQuicksort(A,q+1,r)
```

Stwierdzenie Średni czas sortowania szybkiego z pseudolosowym wyborem elementu wyznaczającego podział jest $\Theta(n\,lg\,n)$, gdzie n to rozmiar sortowanej tablicy.

Idea dowodu
Załóżmy, że elementy tablicy są parami różne. (Bez tego założenia stwierdzenie też jest prawdziwe.)

Niech $T(n)$ oznacza średni czas `RandomizedQuicksort` na n-elementowej tablicy.

$$T(n) \;=\; \Theta(n) + \frac{1}{n}(\sum_{k=1}^{n-1}(T(k)+T(n-k)) + T(n-1) + T(1))$$

Można sprawdzić przez indukcję ze względu na n, że

$$T(n) \leq a\,n\lg n + b$$

dla pewnych stałych a,b

Wybór w oczekiwanym czasie liniowym

```
RandomizedSelect(A,p,r,i)
  if p==r
     return A[p]
  q = RandomizedPartition(A,p,r)
  k = q-p+1
  if i<k
     return  RandomizedSelect(A,p,q,i)
  else
     return  RandomizedSelect(A,q+1,r,i)
```

Stwierdzenie Pesymistyczny czas `RandomizedSelect` jest $\Theta(n^2)$, gdzie n to rozmiar tablicy, z której wybieramy element.

Stwierdzenie Średni czas `RandomizedSelect` jest $\Theta(n)$, gdzie n to rozmiar tablicy.

Uwaga Istnieje algorytm, który wybiera i-ty co dowielkości element z n-elementowej tablicy w pesymistycznym czasie $\Theta(n)$.