**Лекція 5**

**Двійковий (дихотомічний, бінарний) пошук**

Нехай треба з’ясувати, чи є слово Х в словнику, який містить *п* слів. При ***послідовному пошуку*** ми виконаємо, в гіршому випадку, *п* порівнянь. Той самий результат буде й при пошуку в невпорядкованому массиві.

Коли ***пошук деякого елемента*** необхідно здійснити в упорядкованій по зростанню або спаданню послідовності, тоді застосовують ***алгоритм двійкового (бінарного) пошуку***. Метод використовує стратегію «розділяй та господарюй», а саме: задана послідовність ділиться на дві рівні частини й пошук здійснюється в одній з цих частин, яка потім також ділиться навпіл, і так до тих пір, поки з’ясується наявність шуканого елемента або його відсутність. Використання такої операції, яка на кожному кроці зменшує зону пошуку вдвічі, можливе на основі такого факту, що

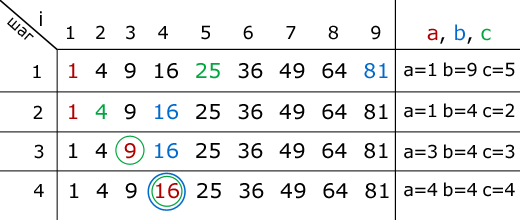
***елементи послідовності були вупорядковані***. Знайшовши середній елемент (сделать это, зная число элементов массива, не составит труда), й порівнявши його значення з шуканим, можна впевнено сказати, де відносно середнього елемента знаходиться шуканий елемент.

Далі, будемо вважати, що елементи масиву розташовані в порядку зростання, оскільки  немає суттєвої різниці, як саме вони впорядковані: по зростанню або спаданню значення. Також позначимо границі зони пошуку через left та right – крайній лівий та крайній правиый елементи відповідно.

Хід роботи алгоритма, розділений на етапи, виглядає наступним чином:

1. зона пошуку (на першому кроці це увесь масив) ділиться на дві рівні частини шляхом знаходження середнього (mid) елемента;
2. середній елемент порівнюється з шуканим (key), результатом чого буде один з трьох випадків:
   * + key<mid. Крайньою правою границею області пошуку стає елемент, який стоїть перед середнім (right ← mid-1);
     + key>mid. Крайньою лівою границею області пошуку стає наступний за середнім елемент (left ← mid+1);
     + key=mid. Значення середнього й шуканого елементів співпадають, отже, елемент знайдено, работа алгоритма завершується.
3. якщо для перевірки не залишилось елементів, то алгоритм завершується, інакше виконується перехід до пункту 1.

В таблиці нижче представлено конкретний цілочисловий масив, й покрокове виконання алгоритма бінарного пошуку. Для економії місця в таблиці left, right і mid замінено на a, b і c.

[](http://kvodo.ru/wp-content/uploads/SearchBinaryAlgorithml.png)

Є послідовність цілих чисел, розташованих у порядку зростання, а також ключ – число 16. Спочатку граничними елементами є елементи з номерами 1 та 9 і значеннями 1 та 81. Обчислюється номер середнього елемента за формулою (right+left)/2 (або left+(right-left)/2 ). Після порівняння маємо, що шуканий елемент менший за середній, тому подальший пошук здійснюється в лівій частині послідовності. На четвертому кроці алгоритм знаходить шуканий елемент.

**Підрахунок числа кроків для довільного *п* (довжина масива)**

 ,

,

але к – натуральне, тому .

Відмітимо, що тут знадобиться набагато менше часу, ніж, якщо б ми скористались лінійним пошуком, але на відміну від лінійного пошуку двійковий працює тільки з впорядкованими масивами, що, безумовно, є недоліком.

***Алгоритм* бінарного піднесення до степеня**

При звичайному підході треба n множень

Зауважимо, що для будь-якого числа a й парного числа n виконується очевидна тотожність

 a^n = (a^{n/2})^2 = a^{n/2} \cdot a^{n/2} 

Вона і є основною в методі бінарного піднесення до степеня. Дійсно, для **парного** n ми показали, як, всього за одну операцію множення, можна звести задачу до вдвічі меншого степеня.

Залишилось зрозуміти, що робити, якщо показник n степеня є **непарним**. Тут ми спочатку переходимо ло степеня n-1, який буде уже парним:

 a^n = a^{n-1} \cdot a 

**Підрахунок числа кроків:**

**кращий варіант:** *п* – степінь двійки, тоді , тобто знову 

**найгірший варіант**: *п* – непарне, тоді може додатись *р<п* кроків переходу від непарних степенів до парних, що суттєво не ускладнює алгоритм.

