Линеарно претраживање - итеративна функција

```
/* linsearch: nalazi vrednost x u nizu b[1], . . . , b[m]  */
int linsearch(int x, int b[ ], int m)
{
    int i=0;
    do i++; /*  b[j] != x za j=1..i-1  */
    while(i<=m && b[i]! = x);
    if (i==m+1) return -1;
    else return i;
}  /*  (b[j] != x za j=1..i-1 && b[i] == x)  */
                    /*  || b[j] != x za j=1..m  */
```

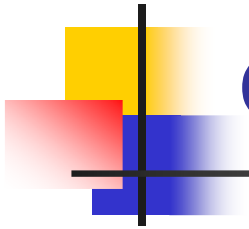
Линеарно претраживање – комплетније решење

```
/*  linsearch: nalazi vrednost x u nizu b[1], . . . , b[m]  */
/* niz b se prosiruje m+1.vim clanom jednakim x  */
int linsearch(int x, int b[ ], int m)
{
    int i=0;
    b[m+1] = x;
    do i++; /*  b[j] != x za j=1..i-1  */
    while(b[i]! = x); /*  (b[j] != x za j=1..i-1 && b[i] == x)  */
    return i;
}
```

Линеарно претраживање – рекурзивна функција

```
int linsearch(int x, int b[], int i, int n)
{
    if(i>n) return -1;
    if (b[i]==x)return i;
    return linsearch(x, b, i+1, n);
}
```

Линеарно претраживање – сложеност



Временска: O(n)

Просторна O(1) / O(n)

Бинарно претраживање: итеративна функција

```
/*  binsearch: nalazi vrednost x u nizu b[1], . . . , b[m]; */
int binsearch(int x, int b[ ], int m)
{
    int l=1, d=m, s;
    while (l <= d) {
        s = (l + d) / 2;
        if (x < b[s]) d = s - 1;
        else if (x > b[s]) l = s + 1;
        else return s;
    }
    return -1;
}
```

Бинарно претраживање - сложеност

- Ако са $S(m)$ ообележимо број потребних операција да се претражи низ од m елемената, онда је
- $S(m) = 1 + S(m/2)$
- Ако за m узмемо само бројеве облика 2^M, онда за такве бројеве важи
- $S(2^M) = 1 + S(2^{M-1}) = 2 + S(2^{M-2}) = \ldots = M + S(2^0) = M + 1.$
- Imamo da je $M = log_2 m$, tj. $S(m) \sim log_2 m$, sto je funkcija koja mnogo sporije raste od m.
- Kako je $\log_e m \sim log_2 m$, to je i $S(m) \sim \ln m$.

Бинарно претраживање – рекурзивна функција

```
int binsearchr(int x, int b[], int l, int d)
{
    int s;
    if(l>d) return -1;
    else{
        s=(l+d)/2;
        if(x==b[s]) return s;
        else if(x<b[s]) d=s-1;
        else l=s+1;
        return binsearchr(x,b,l,d);
    }
}
```

Интерполирано претраживање

- Модификација бинарног:
- Уместо s=(l+d)/2 (s= l+(1/2)*(d-l)),
- може да се користи, на пример,
- s=l+(x-b[l])*(d-l) / (b[d] - b[l])
- Пример:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
P	R	I	M	E	R	P	R	E	T	R	A	Ž	I	V	A	NJ	A

- Сортирани низ:

A	A	A	E	E	I	I	M	NJ	P	P	R	R	R	R	T	V	Ž

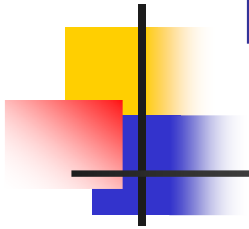
Интерполирано претраживање: пример

- Нека се тражи елемент са вредношћу V:
- Бинарно претраживање:
- A A A E E I I M NJ P P R R R R T V
- P P R R R R T V Z
- R T V Z
- V Z

- Интерполирано претраживање:

ind.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
el.		A	A	A	E	E	I		I	M	NJ	P		P	R	R	R	R	T	V	Ž
kod:	1	1	1	9	9	13	13	18	20	22	22	23	23	23	23	26	28	30
																	V	Ž

Интерполирано претраживање: сложеност



- loglog *n*
- Interpolation Search - A LogLogN Search
- http://www.cs.technion.ac.il/~itai/publications/Algorithms/p550-perl.pdf