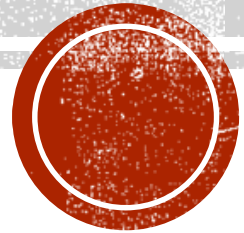
ELEMI-ALGORITMUSOK

függvények / eljárások



LINEÁRIS KERESÉS

```
int lineariskereses(int *a, int n, int szam){
//megkeresi az a címen kezdődő n elemű számsorozatban
//a szam értéket, és visszatéríti az indexét

  for ( int i = 0 ; i < n ; ++i ){
    if ( a[i] == szam ){
      return i;
    }
  }
  return -1; //amennyiben nincs szam értékű eleme a számsorozatnak
}
```

BINÁRIS KERESÉS

```
int binariskereses(int *a, int n, int szam){
  int ah = 0, fh = n - 1, kozep;
  while( ah <= fh ){
    kozep = ( ah + fh ) / 2;
    if ( a[kozep] == szam ){
      return kozep;
    }
    else if ( a[kozep] > szam ){
      fh = kozep - 1;
    }
    else {
      ah = kozep + 1;
    }
  }
  return -1;
}
```

BINÁRIS KERESÉS (REKURZÍVAN)

```
int binariskereses(int *a, int ah, int fh, int szam){
//megkeresi az a[ah..fh] számsorozat-szakaszon a szam értéket

  if ( ah > fh ){ return -1; }
  int kozep = ( ah + fh ) / 2;
  if ( a[kozep] == szam ){ return kozep; }
  if ( a[kozep] > szam ){
    return binariskereses (a, ah, kozep - 1, szam);
  }
  else {
    return binariskereses (a, kozep + 1, fh, szam);
  }
}
```

MAX/MIN-KERESÉS

```
int maxindexkereses(int *a, int n){
//visszatéríti az a címen kezdődő n elemű számsorozat
//legnagyobb elemének indexét

  int maxindex = 0;
  for ( int i = 1 ; i < n ; ++i ){
    if ( a[i] > a[maxindex] ){
      maxindex = i;
    }
  }
  return maxindex;
}
```

MAX/MIN-KERESÉS (REKURZÍVAN)

```
int* maxkereses(int *a, int n){
//visszatéríti a max-érték címét

  if ( n == 1) { return a; }
  else {
    int *p = maxkereses(a+1, n-1);
    if ( *a > *p ) { return a; }
    else { return p; }
  }
}
```

MEGSZÁMLÁLÁS/LÉTEZÉS-ELLENŐRZÉS

```
int darabszam(int *a, int n, int szam){
//visszatéríti, hogy hány szam értékű eleme van
//az a címen kezdődő n elemű számsorozatnak

  int db = 0;
  for ( int i = 0 ; i < n ; ++i ){
    if ( a[i] == szam ){
      ++db; //return 1; (amennyiben létezésellenőrzésről lenne szó)
    }
  }
  return db; //return 0; (amennyiben létezésellenőrzésről lenne szó)
}
```

LÉTEZÉS-ELLENŐRZÉS (REKURZÍVAN)

```
bool letezik_e(int *a, int n, int szam){
//aszerint térít vissza true/false értéket, hogy az a címen
//kezdődő n elemű számsorozat tartalmazza-e a szam értéket

  if ( n == 0 ) { return false; }
  if ( *a == szam ) { return true; }
  return letezik_e(a+1,n-1,szam);
}
```

ÖSSZEG/SZORZAT/ÁTLAG

```
double atlag(int *a, int n){
//visszatéríti az a címen kezdődő n elemű számsorozat
//elemeinek átlagát

  int osszeg = 0; //szorzat = 1; (amennyiben szorzatszámításról lenne szó)
  for ( int i = 0 ; i < n ; ++i ){
    osszeg += a[i]; //szorzat *= a[i]; (amennyiben szorzatszámításról lenne szó)
  }
  return (double)osszeg / n;
}
```

SZÉTVÁLOGATÁS

```
int szetvalogatas (int *a, int n){
//Az a címen kezdődő n elemű számsorozat elemeit szétválogatja oly módon,
//hogy az a[0]-nál kisebbek eléje, a nagyobbak pedig mögéje kerüljenek

  int i = 0, j = n-1, v, x = a[0];
  while ( i < j ){
    if ( a[i] > a[j] ) { v = a[i]; a[i] = a[j]; a[j] = v; }
    else if ( a[i] == x ) { --j; }
    else { ++i; }
  }
  return i;
}
```

ÖSSZEFÉSÜLÉS

```
void osszefesules (int *a, int n, int *b, int m, int *c){
//Az a[0..n-1] és b[0..m-1] számsorokat összefésüli a c[0..n+m-1] tömbbe

  int i, j, k;
  i = j = k = 0;
  while ( i < n && j < m ){
    if ( a[i] < b[j] ) { c[k++] = a[i++]; }
    else { c[k++] = b[j++]; }
  }
  while ( j < m ){
    c[k++] = b[j++];
  }
  while ( i < n ){
    c[k++] = a[i++];
  }

}
```

KIVÁLASZTÁSOS RENDEZÉS

```
void kivalasztorendezes(int *a, int n){
//rendezi az a[0..n-1] számsorozatot

  for( int i = 0 ; i < n-1 ; ++i ){
    int minpoz = i;
    for( int j = i + 1 ; j < n ; ++j ){
      if ( a[minpoz] > a[j] ) { minpoz = j; }
    }
    int veder = a[minpoz];
    a[minpoz] = a[i];
    a[i] = veder;
  }
}
```

BUBORÉK RENDEZÉS

```
void buborekrendezes(int *a, int n){
//rendezi az a[0..n-1] számsorozatot

  do{
    int utolsocserepoz = -1;
    for( int i = 0 ; i < n-1 ; ++i ){
      if ( a[i+1] > a[i] ) {
        int v = a[i]; a[i] = a[i+1]; a[i+1] = v;
        utolsocserepoz = i;
      }
    }
    n = utolsocserepoz + 1;
  }while( utolsocserepoz != -1 );
}
```

BESZŰRÓ RENDEZÉS

- Lásd google…