F.CS203 - Өгөгдлийн бүтэц ба алгоритм 2022-2023

Д.Золбоо

МТ-н салбарын багш

Эрэмбэлэлт

- Bubble Sort – Бөмбөлгөн эрэмбэлэлтийн арга
- Selection Sort – Сонгон эрэмбэлэх арга
- Insertion Sort – Оруулан эрэмбэлэх арга
- Index Sort – Индексээр эрэмбэлэх арга
- Quick Sort – Хурдан эрэмбэлэлтийн арга
- Merge Sort – Нийлүүлэн эрэмбэлэх арга
- Binary Tree Sort - Хоёртын модны эрэмбэлэлт
  Heap Sort – Heap эрэмбэлэлтийн арга

Оршил

- Элементүүдийн эрэмбэлэгдсэн дараалал нь ямар нэгэн элемент хайх процессийг хөнгөвчилж өгдөг. Ж нь: Утасны жагсаалтаас утасны дугаар хайх, Ангийн сурагчдын жагсаалтаас оюутан хайх. г.м. Ингэж эрэмбэнд оруулахыг эрэмбэлэлт гэдэг.

Оршил

- Хэрэв тухайн файлын дурын i < j дугаарын хувьд k[i] элемент тодорхой түлхүүрийн дагуу k[j] элементийн өмнө байрлаж байвал тухайн файл эрэмбэлэгдсэн гэж үзнэ. Эрэмбэлэлтийн түлхүүр нь ихэвчлэн цагаан толгойн үсгийн дарааллаар эрэмбэлдэг
- Ж нь: оюутны бичлэг id, нэр, мэргэжил, gpa гэсэн талбаруудтай бөгөөд id болон нэр нь эрэмбэлэлтийн түлхүүр байж болох юм.

Төсөөлөл

Анхаарч үзэх хүчин зүйлс:

- Програмыг гүйцэтгэхэд хэрэгтэй цаг хугацаа.
- Програмд шаардлагатай санах ойн хэмжээ.
- Ихэвчлэн сонгодог эрэмбэлэлтийн алгоритмууд O(n log n) -> O(n2)хугацаа шаарддаг

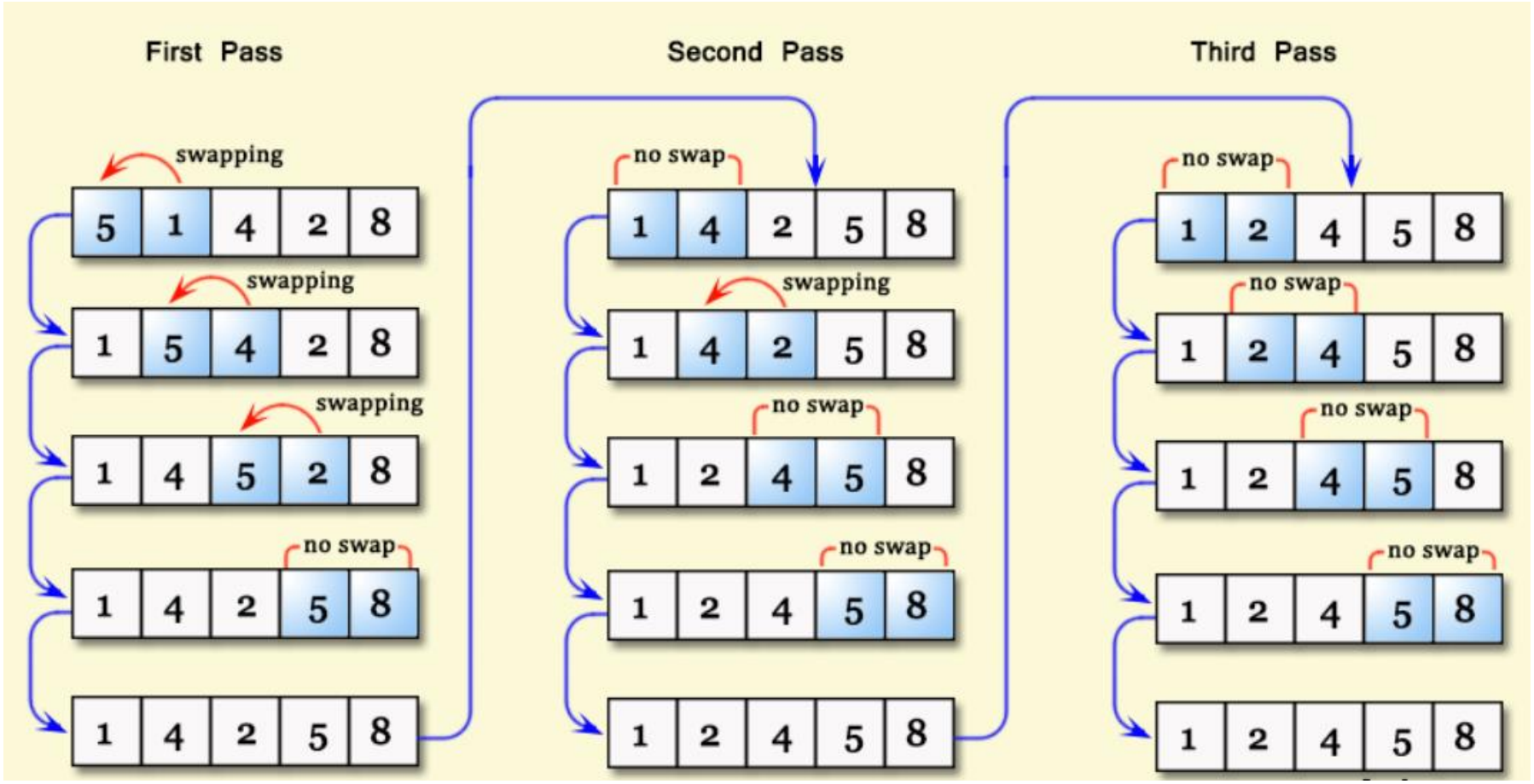
Bubble Sort – Бөмбөлгөн эрэмбэлэлтийн арга

- Үндсэн санаа нь файлын дагуу хэд хэдэн удаа дэс дараалан нэвтрэх явдал юм.Нэвтрэлт бүр нь файл дахь элемент бүрийг хооронд нь харьцуулж шаардлагатай тохиолдолд байрыг нь солих бөгөөд энэ бүгдийн эцэст хамгийн их (бага) элемент зөв байрлалдаа орох болно.

Bubble Sort – Бөмбөлгөн эрэмбэлэлтийн арга

- i итерац хийгдсэний дараа n-i их буюу тэнцүү байрлалд байгаа элементүүд зөв байрлалдаа орсон байна.
- n хэмжээтэй файл эрэмбэлэхэд n-1 нэвтрэлт хэрэгтэй. Энэнээс ч эрт эрэмбэлэгдсэн байж болно.

Bubble Sort – Бөмбөлгөн эрэмбэлэлтийн арга



Bubble Sort – Бөмбөлгөн эрэмбэлэлтийн арга

```
void bubble (ElementType x[], int n) {
  int hold, j, pass;
  bool switched = true;
  /* гадна талын давталт нэвтрэлтийн тоог хянана */
  for (pass=0; pass < n-1 && switched==true; pass++) {
    /* эхэндээ ямар нэгэн байр солилт хийгдээгүй гэж үзнэ */
    switched = false;
    for (j = 0; j < n-pass-1; j++)
```

Bubble Sort – Бөмбөлгөн эрэмбэлэлтийн арга

```
    /* дотоод давталт элементүүдийг харьцуулна */
    if (x[j] > x[j+1]) {
      /* Байр солилт хийх шаардлагатай бол */
      switched = true;
      hold = x[j];
      x[j] = x[j+1];
      x[j+1] = hold;
    } /* end if */
  } /* end for */
} /* end bubble */
```

Big-Oh O тэмдэглэгээ

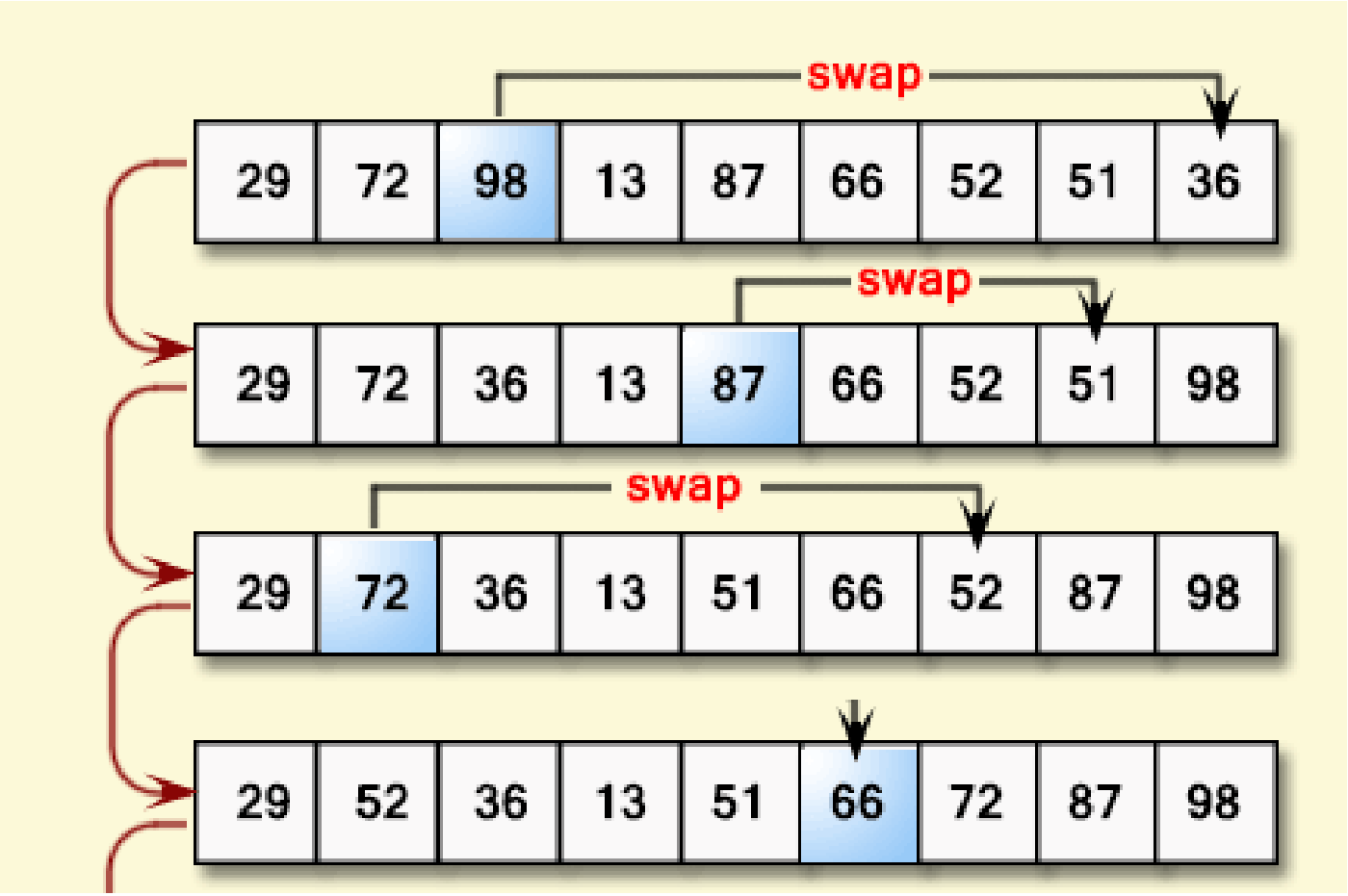
- n-1 нэвтрэлт ба n-1 харьцуулалт хийгдэнэ. (n-1)*(n1) = n^2 - 2n + 1 Эндээс O(n^2).

- i дүгээр итерац дээрх харьцуулалтын тоо нь n-i. k итерацийн нийт харьцуулалтын тоо нь: (n-1)+(n-2)+ ... +(n-k) ба эндээс (2kn-k^2 -k)/2. к итерацийн дундаж харьцуулалт O(n), боловч нийт эрэмбэлэлтийн хугацаа нь O(n^2) хэвээрээ байна.

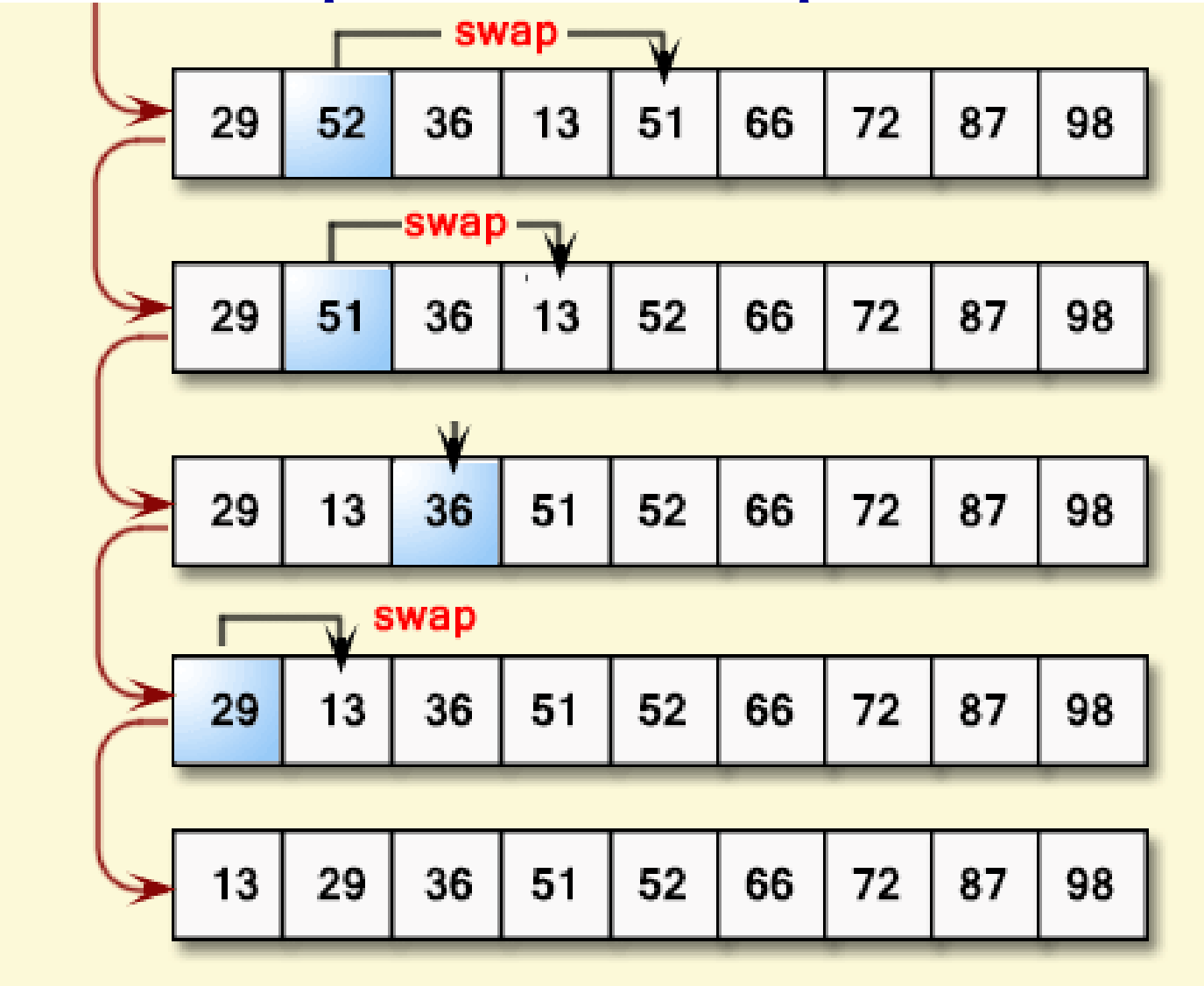
Selection Sort Сонгон эрэмбэлэх арга

- Сонгон эрэмбэлэх арга нь шаардлагатай элементийг сонгон авч зөв байрлалд нь оруулах арга юм.
- Өгөгдсөн х массив эрэмбэлэгдээгүй байг.
- Үлдсэн элементүүдээс хамгийн ихийг нь large хувьсагчид авна, Дараа нь массивийн төгсгөлийн элементтэй large хувьсагчийг байрыг нь солино

Selection Sort Сонгон эрэмбэлэх арга



Selection Sort Сонгон эрэмбэлэх арга



Selection Sort Сонгон эрэмбэлэх арга

```
void selectsort(ElementType x[], int n) {
  int i, indx, j, large;
  for (i = n-1; i > 0; i--) {
    /* place the largest number of x[0] through */
    /* x[i] into large and its index into indx */
    large = x[0];
    indx = 0;
```

Selection Sort Сонгон эрэмбэлэх арга

```
  for (j = 1; j <= i; j++)
    if (x[j] > large) {
      large = x[j];
      indx = j;
    } /* end for ... if */
    x[indx] = x[i];
    x[i] = large;
  } /* end for */
} /* end selectsort */
```

Selection Sort Сонгон эрэмбэлэх арга

- Хамгийн эхний итерац n-1 харьцуулалт хийнэ, дараагийнх нь n-2, гэх мэтчилэн. , (n-1) +(n-2) + (n-3) + ... + 1= n*(n-1)/2 буюу O(n2).
- Энд ямар нэгэн сайжруулалт байхгүй.

Insertion sort оруулан эрэмбэлэх арга

- Оруулан эрэмбэлэх арга нь N-1 итерацаас тогтоно.
- p=1 -> N-1 хүртэл итерацид, 0 -> p хүртэл элементүүд эрэмбэлэгдсэн гэж үзнэ.

Insertion sort оруулан эрэмбэлэх арга

25	57	48	37	12	92	86	33	0
25	57	48	37	12	92	86	33	1
25	48	57	37	12	92	86	33	2
25	37	48	57	12	92	86	33	3
12	25	37	48	57	92	86	33	4
12	25	37	48	57	92	86	33	5
12	25	37	48	57	86	92	33	6
12	25	33	37	48	57	86	92	7

Insertion sort оруулан эрэмбэлэх арга

```
void InsertionSort( ElementType a[], int n ) {
        int j, p;
        ElementType tmp;
        /* 1*/
        for( p = 1; p < n; p++ ) {
                /* 2*/ tmp = a[ p ];
                /* 3*/ for( j = p; j > 0 && tmp < a[ j - 1 ]; j–
                        /* 4*/ a[ j ] = a[ j - 1 ];
                /* 5*/ a[ j ] = tmp;
        }
}
```

Insertion Sort-ын үр ашиг

- Хэрэв оролтын массив эрэмбэлэгдсэн бол итерац бүрт зөвхөн ганц харьцуулалт хийгдэнэ. Иймм учраас ажиллах хугацаа нь O(N) болно.
- Хэрэв оролтын массив урвуу дарааллаар эрэмбэлэгдсэн бол ажиллах хугацаа нь $O(n^2)$, Нийт харьцуулалтын тоо нь: (n-1) +(n-2) + (n-3) + ... + 1= n*(n-1)/2
- Энгийн оруулан эрэмбэлэх арга нь ихэнх тохиолдолд бөмбөлгөн эрэмбэлэлтийн аргаас илүү байдаг. Бараг эрэмбэлэгдсэн массивийн хувьд илүү үр ашигтай болно.

Insertion Sort хугацаа

- Оруулан эрэмбэлэх аргын харьцуулалтын дундаж хугацаа нь $O(n^2)$. Шаардагдах зай нь нэг хувьсагчийн зай болно.
- Ердийн оруулан эрэмбэлэх арга нь бараг эрэмбэлэгдсэн массивийн хувьд өндөр үр ашигтай байдаг. Жижигхэн массивийн хувьд O(n log n) алгоритмаас илүү үр ашигтай ажилладаг.

Index Sort - Индексээр эрэмбэлэх арга

- Зарим тохиолдолд өгөгдлүүд маш том бичлэгээс тогтож байвал эрэмбэлэхэд зарцуулах хугацааны дийлэнх хэсэг нь солих үйлдэлд зарцууллагддаг. Иймт охиолдолд индексийн массив үүсгэж, өгөгдөлд өөрчлөлт оруулахгүйгээр элементүүдийн индексийг уг өгөгдөл эрэмбэлэгдсэн байхаар индексийн массивт хийнэ.
- Эрэмбэлэлтийн энэ аргын үед өгөгдөл хадгалах a[] массиваас гадна индекс хадгалах p[] массив хэрэгтэй. P[] массив нь өгөгдлийн массив дах өгөгдлүүдийн индексийг хадгалах ба индексийн массивт байрлах дарааллаар өгөгдлийн массивт хандвал эрэмбэлэгдсэн өгөгдлүүд гарна. Өөрөөр хэлбэл p[1] нь а массивийн хамгийн бага элементийн индексийг, p[2] нь удаах элементийг гэх мэтчилэн хадгална.

Index Sort - Индексээр эрэмбэлэх арга

Эрэмбэлэлт нь маш том бичлэгүүдийг шилжүүлэн солихын оронд зөвхөн индексүүдийн хооронд солих үйлдэл хийдэг. Индексийн массив үүсгэхдээ өмнө үзсэн аль ч аргыг ашиглаж болно. Дараах жишээнд оруулан эрэмбэлэх аргыг ашигласан.

```
Void indexSort(Type  a[],int p[], int n)
{ int i,j; Type t;
  for(i=0;i<n;i++) p[i]=i;
  for(i=1;i<n;i++) {
    t=p[i]; j=i-1;
    while(j>=0 &&a[t]<a[p[j]]) p[j+1]=p[j--];
    p[j+1]=t;
}
```

Index Sort - Индексээр эрэмбэлэх арга

Индекслэгдэхийн өмнө

A	S	O	R	T	I	N	G	E	X	A	M	P	L	E
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Индекслэгдсэний дараа

A	S	O	R	T	I	N	G	E	X	A	M	P	L	E
0	10	8	14	7	5	13	11	6	2	12	3	1	4	9

Эрэмбэлсний дараа

A	A	E	E	G	I	L	M	N	O	P	R	S	T	X
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14