ТЕМА 3. Алгоритмы сортировки

3.1. Сортировка массивов

Критерии эффективности сортировки :

1. Скорость сортировки.

2. Естественность сортировки.

3. Устойчивость сортировки.

Три основных класса сортировок:

1. Обменые.

2. Выбором.

3. Вставкой.

3.2. Простые методы сортировки

3.2.1. Метод пузырька

```
void s_puz(tmas a[], int n)   {
    tmas t;
   for (int i = 1; i < n; i++)
     for (int j = n - 1; j >= i; j--)
       if (a[j - 1].key > a[j].key)
          {
             t = a[j - 1];
             a[j - 1] = a[j];
             a[j] = t;
          }
}
```

3.2.1. Сортировка выбором

```
void s_vb(tmas a[], int n) {
   int imin, i, j;      tmas t;
  for (i = 0; i < n - 1; i++)  {

      imin = i;
    for (j = i + 1; j < n; j++)
       if (a[imin].key > a[j].key) imin = j;

if (imin != i)  { t = a[imin];  a[imin] = a[i];  a[i] = t; }

                                    }
}
```

3.2.3. Сортировка вставками

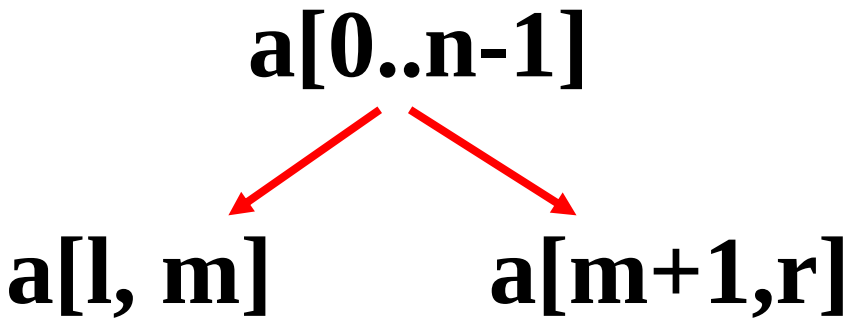
```
void s_vst(tmas a[], int n)  {
tmas t;
   for (int  i = 1; i < n; i++) {
      t = a[i];
      for (int  j = i - 1; j >= 0 && t.key < a[j].key; j--)
                                                a[j + 1] = a[j];
      a[j + 1] = t;
   }
}
```

3.3. Улучшенные методы сортировки

3.3.1. Метод Шелла

```
void s_shell(tmas a[], int n)  {
tmas t;
    for (int d = 3; d > 0; d--)
        for (int  i = d; i < n; i++)  {
            t = a[i];
     for (int  j = i - d; j >= 0 && t.key < a[j].key; j -= d)
                                                    a[j + d] = a[j];
            a[j + d] = t;
        }
}
```

3.3.2. Сортировка слиянием



```
// Функция слияния
void slip(int left, int m, int right) {
   int i = left, j = m + 1, k = 0;

   while ((i <= m) && (j <= right))
      if (a[i].key < a[j].key) { c[k++] = a[i++]; }
                               else { c[k++] = a[j++]; }

   while (i <= m)    c[k++] = a[i++];

   while (j <= right) c[k++] = a[j++];

   // Запись отсортированного участка в массив
   k = 0; i = left;
   while (i <= right) a[i++] = c[k++];
}
```

```
// Функция сортировки
void s_sl(int left, int right) {
    if (left < right) {
        int m = (left + right) / 2;
          s_sl(left, m);
          s_sl(m + 1, right);
            slip(left, m, right);
                        }
}
```

Вызов:

```
s_sl(0, n - 1);
```

3.3.3. Метод QuickSort (быстрая сортировка, сортировка Хоара)

```
struct St { int l;  int r; }   stack[10];

void push(int l, int r, int& s) {
stack[s].l = l;
stack[s].r = r;
s++;
}

void pop(int& l, int& r, int& s) {
s--;
l = stack[s].l;
r = stack[s].r;
}
```

```
void s_qs(tmas a[], int n) {
  int i, j, left, right, s = 0;
  tmas t, x;
     push(0, n - 1, s);
 while (s != -1) {
    pop(left, right, s);
          while (left < right) {
           i = left; j = right; x = a[(left + right) / 2];
             while (i <= j) {
                while (a[i].key < x.key) i++;
                while (a[j].key > x.key) j--;
              if (i <= j) { swap(a[i], a[j]); i++;  j--; }          }
```

```
if ((j - left) < (right - i)) { // Выбор короткой части
    if (i < right) push(i, right, s);
      right = j;
                              }
                        else {
    if (left < j) push(left, j, s);
      left = i;
                              }
}
}
}
```

3.4. Выбор метода сортировки