Опубликовано 13.07.2017
http://www.ollejnik.com

АЛГОРИТМЫ ПАКЕТНОЙ СОРТИРОВКИ МАССИВОВ - серии vladsort или МОЖНО ЛИ ОБОГНАТЬ qsort?

Да, можно. Это ответ на вопрос, вынесенный в заголовок этой статьи.
И возможно, один из предложенных здесь алгоритмов (vladsort_3), может оказаться САМЫМ БОЛЬШИМ и в тоже время САМЫМ БЫСТРЫМ алгоритмом сортировки массивов.

В ЭТОЙ СТАТЬЕ ОПУБЛИКОВАНЫ ДВА ИСХОДНЫХ КОДА АЛГОРИТМОВ СОРТИРОВКИ МАССИВОВ - vladsort_1 И vladsort_2.
ТРЕТИЙ АЛГОРИТМ - vladsort_3 – РАЗМЕЩЁН ПО АДРЕСУ: http://ollejnik.com/files/algoritm-paketnoy-sortirovki-massivov-2.pdf

Все три предложенные алгоритма, являются гибридом сортировки подсчётом и сортировки выбором.
Один из предлагаемых здесь алгоритмов сортировки (vladsort_3) - во многих случаях, на массивах случайных чисел - обгоняет быструю сортировку (qsort) на 5-30%.
Но время сортировки vladsort_3 очень сильно зависит от значения (числа) присваемого переменной "t". При одних значениях "t" - время сортировки может быть медленнее qsort и значительно медленнее.

НО ПРИ ОПРЕДЕЛЁННЫХ ЗНАЧЕНИЯХ ЧИСЛА "t" - vladsort_3 - СОРТИРУЕТ БЫСТРЕЕ qsort.

Что значит число "t" - это величина "кусочка", пакета, отрезка чисел в массиве, которые сортируются в определённом диапазоне.
Значение переменной "t" может равняться от 1 и больше, целым числом.
Например, в алгоритме vladsort_3, при значении t = 1, за один проход по массиву будет отсортировано сразу 1024 чисел. При t = 2 - 2048 чисел. При t = 10 - 10240, при t = 100 - 102400(!) чисел.

От чего зависит число "t" ?
Зависит от нескольких параметров:
1. от величины массива.
2. от величины между минимальным и максимальным числом.
3. от "плотности" чисел в массиве.
4. От некоторых други значений.
Так вот, на разных массивах есть "своё", оптимальное значение числа "t", при котором vladsort_3 обгоняет qsort. И это оптимальное число, так сказать "золотую середину", я получал экспериментальным путём.
Как получать значение "t" путём вычислений, зная выше приведенные параметры массива - пока не знаю.

Но обовсём по порядку.

****

ВНИМАНИЕ! Алгоритмы опубликованы "как есть". Поэтому автор не несёт ответственности за возможно, не корректную их работу и связанные с этим последствия.
Алгоритмы написаны НЕ ПРОФФЕСИОНАЛОМ, в них возможны лишние переменные и лишние некоторые вычисления а так же, могут быть, не замеченные ошибки.

Все алгоритмы и фрагменты алгоритмов представленных в этой статье, написаны на языке C++.

Пока что, уровень моих знаний не позволяет написать программу более изыскано, поэтому, прошу прощения за возможно корявый код.

ПРЕДЫСТОРИЯ СОЗДАНИЯ АЛГОРИТМА.

Как то, примерно полгода назад, сравнивая разные алгоритмы сортировки, я обратил внимание на одну деталь. А именно - малые и очень малые массивы, многие алгоритмы сортируют почти с одинаковой скоростью. А вот средние, большие и очень большие массивы сортируются совсем по разному и с большой разницей во времени сортировки. Алгоритмы сортировки имеют как линейное так и квадратичное время сортировки.
Например, мной измерена скорость сортировки двух алгоритмов - qsort (быстрая сортировка) и сортировка пузырьком. **ВСЕ измерения проводились на одном и том же мощном компьютере.**
И разница во времени сортировки на массиве случайных чисел - оказалась громадной.

(Числа получены с помощью функции rand(), если не указано другое).

Может возникнуть вопрос, КАК ИЗМЕРЯЛОСЬ ВРЕМЯ СОРТИРОВКИ? Очень просто: с помощью цикла for или метки делаем цикл на 1000 (или 100, 10 000 или другое значение) итераций. Тело алгоритма, вместе с функцией rand() помещаем в цикл. Время измеряем с помощью обыкновенной секундной стрелки. Потом делим на количество итераций.
Таким путём получаем СРЕДНЕЕ ВРЕМЯ ИЗ 1000 СОРТИРОВОК и это время, почти всегда одинаково.
(Там ещё есть время размещения чисел в массив, но при одинаковых условиях, время сортировки измеренное этим способом, зависит только от быстродействия алгоритма).

Но думаю, почти все алгоритмы сортируют практически с одинаковой скоростью массив - ИЗ ДВУХ чисел.
И тогда я подумал - "А что, если большой массив разбить на маленькие частично отсортированные "кусочки", а уже их, быстро отсортировать". Эта идея не нова, таких алгоритмов существует немало.

НО, СМЫСЛ В ТО, ЧТО БЫ ВНАЧАЛЕ ПРОВЕСТИ КАК БЫ ГРУБУЮ СОРТИРОВКУ МАССИВА, А УЖЕ ПОСЛЕ, ПРОВЕСТИ ТОНКУЮ СОРТИРОВКУ.
Что из этого получилось, читайте дальше.

НАЧАЛОМ ВСЕХ ПРЕДЛОЖЕННЫХ АЛГОРИТМОВ, ЯВЛЯЕТСЯ - СОРТИРОВКА ВЫБОРОМ.

Здесь представлены ДВА алгоритма - более медленных и ОДИН (размещён на дпугой странице), самый последний и самый быстрый, обгоняющий qsort, по времени сортировки (только при определённом значении "t", которое, для каждого массива, может быть разным). И назвал я их - vladsort (есть же timsort).

"Родителем" всех трёх алгоритмов является, всем известная сортировка выбором.

Привожу фрагмент кода сортировки выбором (одна из версий):

```
for (mas_vel = 100000; mas_vel > 0; mas_vel--) {
                macs = massiv[0];
                for (c = 1; c < mas_vel; c++) if (macs < massiv[c]) macs = massiv[c];
                for (c = 0; c<mas_vel; c++) if (macs == massiv[c]) b = c;
                massiv[b] = massiv[mas_vel - 1];
                massiv[mas_vel - 1] = macs;
        }
```

Как видно из фрагмента, вначале находим самое большое число. Потом находим номер ячейки в которой находится самое большое число. А потом перемещаем в конец массива самое большое число, а число которое находилось в самой последней ячейке массива - перемещаем в ячейку где находилось самое большое число. И эта последовательность операций повторяется до тех пор, пока массив не будет полностью отсортирован.

Сортировка выбором сортирует, примерно в 4 раза быстрее пузырьковой сортировки.
Вот время сортировки пузырьком, выбором и быстрой сортировкой
(получено экспериментально):
массив 100 000 ..... 15 ..... 3,5 ..... 0,020 ....секунд
массив 200 000 .....60 .... 15 ..... 0,040 ...секунд

Как известно, в qsort время сортировки - линейное, в пузырьковой и выбором - квадратичное.
Что значит это на практике, если массив возрастает в 2 раза, время сортировки возрастает - при линейном в 2 раза, при квадратичном в 4 раза. Так же qsort считается самой быстрой сортировкой на сегодняшний день, из всех существующих. А пузырьковая сортировка, одной из самых медленны (есть и более медленные).

Идея заложенная в самый первый написанный мной алгоритм сортировки (vladsort_1), заключается в следующем:
сортировка состоит из ДВУХ ЭТАПОВ.

******* 1 ЭТАП СОРТИРОВКИ *****.**
При каждых двух проходах алгоритма мы находим сразу определённое количество (так сказать, пакет, зависит от числа "t") самых маленьких и самых больших чисел и перемещаем их в начало и в конец алгоритма. Следующий проход алгоритма начинаем с конца первого "пакета" и до начала второго "пакета". Таким путём, предварительная сортировка заканчивается - посередине массива. А массив получаем из "кусочков", набора чисел уже отсортированных в определённом диапазоне.

******* 2 ЭТАП СОРТИРОВКИ *******
С помощью сортировки выбором (фрагмент приведен выше, только "заточен" под данную задачу) сортируем в массиве "кусочки" уже отсортированные в определённом диапазоне.

Вот алгоритм сортировки массивов - vladsort_1 работающий на том же принципе:

```
// Алгоритм сортировки массивов  --  vladsort_1.
#include <iostream>
#include <cstdlib>
using namespace std;

int main()
{
	setlocale(LC_ALL, "Rus");
	int masiv[1000];
	int a, aa, b, c, d, h, s, v, vv, x, y, z, i, xx, t;

	xx = z = 1000;
	t = 10;

	// Набор случайных чисел помещаем в массив.

	for (c = 0; c < z; c++) masiv[c] = rand();

	// Находим минимальное число.

	a = masiv[0];
	for (b = 1; b < z; b++) if (a > masiv[b]) a = masiv[b];

	// Находим максимальное число.

	c = masiv[0];
	for (b = 1; b < z; b++) if (c < masiv[b]) c = masiv[b];

	// Вычисляем сколько надо итераций.

h = c - a;
h = h + 2;
for (; ;) {
	d = h % t;
	if (d == 0) {
		d = h % 2;
		if (d == 0) break;
	}
	h++;
}
b = h / 10;
b = b / 2;
h = h + a;

// Приступаем к сортировке массива (1 этап).

aa = a;
c = h;
s = z;
z--;
y = 0;
for (d = 0; d < b; d++) {
	a = a + t;
	h = h - t;
	v = y;
	for (; v <= z; v++) {
		if (masiv[v] < a) {
			for (; masiv[y] < a;) {
				if (y == v) {
					y++;
					goto num;
				}
				y++;
```

```
                        }
                        x = masiv[y];
                        masiv[y] = masiv[v];
                        masiv[v] = x;
                        y++;
                        continue;
                num:
                        continue;
                }
                if (masiv[v] >= h) {
                        for (; z > v; z--) {
                                if ((z - 1) == v) break;
                                if (masiv[z] < h) break;
                        }
                        x = masiv[v];
                        masiv[v] = masiv[z];
                        masiv[z] = x;
                        z--;
                        if (masiv[v] < a) v--;
                }
        }
}

// Приступаем к сортировке массива (2 этап).

x = s - 1;
b = b * 2;
z = 0;
for (d = 0; d <= b; d++) {
        v = vv = z;
        aa = aa + t;
        for (; v < s; v++) {
                if (masiv[v] < aa) {
                        if (v == x) goto mit;
                        if (masiv[v + 1] >= aa) {
                        mit:
                                if (vv == v) {
                                        z = ++v;
                                        break;
                                }
                                z = v = ++v;
                                for (; v > vv; v--) {
                                        h = masiv[vv];
                                        for (c = (vv + 1); c < v; c++) if (h < masiv[c]) h = masiv[c];
                                        for (c = vv; c < v; c++) if (h == masiv[c]) y = c;
                                        if (masiv[y] == masiv[v - 1]) continue;
                                        masiv[y] = masiv[v - 1];
                                        masiv[v - 1] = h;
                                }
                                break;
                        }
                        continue;
                }
                if (masiv[v] >= aa) break;
        }
}
return 0;
}
```

Этот алгоритм при t = 10 - в несколько раз медленнее быстрой сортировки.

А что если "кусочки" находить и перемещать в начало и в конец массива, не один а 10 или 20, подумал я. И тогда я написал следующий алгоритм, который за 2 прохода находит и перемещает сразу 20 наборов, "пакетов" чисел. Если t = 10, то за два прохода по массиву, алгоритм предварительно отсортирует сразу 400 чисел. 200 с "головы" массива и 200 с "хвоста" массива.

И такой алгоритм был написан, это алгоритм сортировки массивов - vladsort_2.
Этот алгоритм имеет около 1380 строк.
При тестировании данного алгоритма (t=15), оказалось, что на массиве в 50 000 чисел - он опережает быструю сортировку всего на 2%. И этот алгоритм оказался значительно быстрее первого алгоритма.
По скорости сортировки, на некоторых массивах случайных чисел он вплотную подошёл к быстрой сортировке, а на массиве 50 000 чисел - чуть-чуть обогнал быструю сортировку.

Код алгоритма сортировки массивов - vladsort_2 - в конце этой статьи.

И тогда...

И тогда я написал алгоритм сортировки массивов - vladsort_3, который сортирует на первом этапе, сразу 512 "кусочков" - с начала и с конца массива. Это при t = 1. И этот алгоритм по времени, превосходит быструю сортировку (при определённых значениях "t") - до 30% на массивах случайных чисел.
А при t = 10 - сортирует сразу 10240(!) чисел, при t = 20 - 20480 чисел, при t = 100 - дальше подсчитайте сами.
Алгоритм может сортировать от 1024 до сотен тысяч и миллионов чисел (и больше), всего за 2 прохода по алгоритму.

Алгоритмы сортировки массивов - vladsort_2 и vladsort_3 - по структуре и по действиям выполняемым кодом - практически одинаковы. Я даже использовал vladsort_2 - как шаблон, при написании кода алгоритма vladsort_3.

Да, алгоритм сортировки массивов - vladsort_3 - огромен !!

********** vladsort_3 - это:
около 35 000(!) строк исходного кода (возможно и меньше, писал как быстрее).
более 4 096 локальных переменных (512 по восемь раз).
компиляция кода длится более 9 минут.
"вес" исходника - более 900 кб (есть возможность уменьшить).
все 35000 строк умещаются в одной функции.
компилятор зависает на строке "Создание кода", если массив перенести в глобальную область а переменные локальные.
(пока не нашёл решение этого вопроса, надо искать варианты, в частности попробовать другие компиляторы.
Использовал компилятор в IDE - Visual Studio Community 2015).
****

Так же компилятор, ведёт себя как то странно. На ошибки - НЕ РУГАЕТСЯ!! Например, если в одном месте кода вместо переменной h142, будет просто 142, код создастся, как будь-то было h142! Но будут не отсортированные числа, в уже отсортированном массиве. Таким образом переменная "142" будет "восприниматься" компилятором как h142!!
По всей вероятности, идёт переполнение стэка синтаксического анализатора и он попросту входит в "ступор" и перестаёт анализировать..
Мной были, при написании кода, допущены 3 ошибки (одна из них, приведенна выше) - за которыми я несколько дней "бегал", пока не нашёл и исправил.

Ну а теперь ХАРАКТКРИСТИКИ времени сортировки - qsort и vladsort_3
(получены экспериментально):

на массиве "сплошных" чисел, полностью отсортированном в обратном порядке:
(t = 20, число 20 оптимальное для таких массивов)
..... vladsort_3 .....
200 000 .....0,01 сек ..... (qsort - 0,014 сек) ..на 40% быстрее qsort
10 000 .....0,000375 сек
5 000 .....0,000175 сек
1 000 .....0,0000375 сек
~~~~~~

при t = 4
массив .....qsort ..... vladsort_3 .....
200 000 .. 0,040 сек .....0,0365 сек, ...на 10% ...быстрее qsort.
100 000 .. 0,020 сек..... 0,0175 сек, ...на 14% ...быстрее qsort.
.. 50 000 .. 0,0097 сек .....0,00888 сек, ...на 9% ....быстрее qsort.
. 40 000 .. 0,00783 сек .....0,0072 сек, ...на 9% ....быстрее qsort.
. 30 000 .. 0,00575 сек .....0,00543 сек, ...на 6% ....быстрее qsort.
. 20 000 .. 0,00375 сек .....0,0037 сек, ...на 1% ....быстрее qsort.
. 10 000 .. 0,00174 сек .....0,00195 сек, ...на 12% ..МЕДЛЕННЕЕ qsort.
... 5 000 .. 0,00086 сек .....0,001 сек, ...на 16% ..МЕДЛЕННЕЕ qsort.
... 1 000 .. 0,000147 сек ...0,000235 сек ...на 60% .МЕДЛЕННЕЕ qsort.
~~~~~~

..... 100 .. 0,0000114сек .0,0000475 сек .на 317% .МЕДЛЕННЕЕ qsort.

Как видно из замеров, vladsort_3 на массивах от 20 000 и больше - или равняется или опережает быструю сортировку по времени упорядочивания массива.
На массивах менее 20 000, vladsort_3 - медленнее qsort.
НО это, при t = 4.
Если мы возьмем "t" с другим числом, тогда скорость сортировки на меньших массивах, будет другая и она превысит, скорость сортировки qsort.

Например:
t = 80 .....массив 10 000 - 0,00145 сек на 20% быстрее qsort
t = 80 .....массив 20 000 - 0,0033 сек ..на 14% быстрее qsort
t = 80 .....массив 30 000 - 0,0055 сек ..на 4,5% быстрее qsort

Например:
t = 4, ...массив 1 000 - 0,000235 сек. ....на 60% медленнее qsort.
t = 40, ...массив 1 000 - 0,000175 сек ....на 19% медленнее qsort.
t = 300, ..массив 1 000 - 0,00012сек. ....на 23% БЫСТРЕЕ qsort.

ОБЛАСТИ ПРИМЕНЕНИЯ АЛГОРИТМА СОРТИРОВКИ МАССИВОВ - vladsort_3.

В принципе, предлагаемый алгоритм может заменить быструю сортировку. На массивах, от несколько сот чисел и до очень больших.
Ведь на всех массивах, с которыми я экспериментировал, vladsort_3 быстрее qsort (но может и медленнее, например, если массив не большой, а ОДНО число большое или очень большое).
Но это в принципе...
По причине больших габаритов и сильной зависимости скорости сортировки от задаваемого значения переменной t, а так же, не большого выиграша во времени, по сравнению с qsort - на практике, не думаю, что он сможет заменить qsort.
Но в узком плане применения, возможно.
Например, если надо сортировать входные массивы с похожими наборами чисел, когда оптимальное значение t можно более-менее точно указать заранее.

Есть ещё вариант, проводить ПРЕДВАРИТЕЛЬНЫЙ АНАЛИЗ МАССИВА перед сортировкой.
НО...
1. На анализ массива уйдёт некоторое время. Чем детальнее анализ - тем больше времени.
2. На основе каких параметров, проводить анализ массива.
На всё это уйдё дополнительное время, которое возможно, "съест" тот не значительный выигрыш во времени сортировки.
(над этим я думаю, что из этого получится - пока не знаю).

****
****

А сейчас, привожу код алгоритма сортировки массивов - vladsort_2 (исходный код):

```
// Алгоритм сортировки массивов - vladsort_2.
/*
Этот алгоритм сортирует массивы чисел в два этапа.
На первом этапе проводится "грубая" сортировка.
В этом алгоритме, сортируются сразу по 20 "кусочков", пакетов набора чисел  в определённом диапазоне, с начала и с конца массива.
Длина "кусочка" зависит от заданного числа переменной t.

На втором этапе сортируем "кусочки" чисел с помощью сортировки выбора, "заточенной" под такую задачу.
*/

#include <iostream>
#include <cstdlib>
using namespace std;

int main()
{
	setlocale(LC_ALL, "Rus");
	int masiv[50000];
```

```
int a, b, c, d, h, s, v, vv, x, y, z, bb, an, t, i;

int a2, a3, a4, a5, a6, a7, a8, a9, a10, a11, a12, a13, a14, a15, a16, a17, a18, a19, a20;
int h2, h3, h4, h5, h6, h7, h8, h9, h10, h11, h12, h13, h14, h15, h16, h17, h18, h19, h20;
int aa, a2a, a3a, a4a, a5a, a6a, a7a, a8a, a9a, a10a, a11a, a12a, a13a, a14a, a15a, a16a, a17a, a18a, a19a, a20a;
int hh, h2h, h3h, h4h, h5h, h6h, h7h, h8h, h9h, h10h, h11h, h12h, h13h, h14h, h15h, h16h, h17h, h18h, h19h, h20h;
int ah, a2h, a3h, a4h, a5h, a6h, a7h, a8h, a9h, a10h, a11h, a12h, a13h, a14h, a15h, a16h, a17h, a18h, a19h, a20h;
int ha, h2a, h3a, h4a, h5a, h6a, h7a, h8a, h9a, h10a, h11a, h12a, h13a, h14a, h15a, h16a, h17a, h18a, h19a, h20a;
int as, as2, as3, as4, as5, as6, as7, as8, as9, as10, as11, as12, as13, as14, as15, as16, as17, as18, as19, as20;
int az, az2, az3, az4, az5, az6, az7, az8, az9, az10, az11, az12, az13, az14, az15, az16, az17, az18, az19, az20;

z = 50000;
t = 15;

// Набор случайных чисел помещаем в массив.

for (c = 0; c < z; c++) masiv[c] = rand();

// Находим минимальное число.

a = masiv[0];
for (b = 1; b < z; b++) if (a > masiv[b]) a = masiv[b];

// Находим максимальное число.

c = masiv[0];
for (b = 1; b < z; b++) if (c < masiv[b]) c = masiv[b];

// Вычисляем сколько надо итераций.

an = a;
b = c - a;
b = b + 2;
for (; ;) {
        d = b % t;
        if (d == 0) {
                d = b % 2;
                if (d == 0) break;
        }
        b++;
}
h = b;
b = b / t;
bb = b;
b = b / 2;
h = h + a;

//  Вычисляем сколько надо ДВАДЦАТОК.

for (; ;) {
        d = b % 20;
        if (d == 0) break;
        b++;
}
d = b / 20;
s = z;

//  Приступаем к сортировке (1 этап).

//   a20 - это < 10 (или <38).
//   a - это < 200 (или <228).

a20 = a;
y = --z;
v = b = 0;
for (c = 0; c < d; c++) {
```

```
a19 = a20 + t;
a18 = a19 + t;
a17 = a18 + t;
a16 = a17 + t;
a15 = a16 + t;
a14 = a15 + t;
a13 = a14 + t;
a12 = a13 + t;
a11 = a12 + t;
a10 = a11 + t;
a9 = a10 + t;
a8 = a9 + t;
a7 = a8 + t;
a6 = a7 + t;
a5 = a6 + t;
a4 = a5 + t;
a3 = a4 + t;
a2 = a3 + t;
a = a2 + t;
x = a + t;

h = h - t;
h2 = h - t;
h3 = h2 - t;
h4 = h3 - t;
h5 = h4 - t;
h6 = h5 - t;
h7 = h6 - t;
h8 = h7 - t;
h9 = h8 - t;
h10 = h9 - t;
h11 = h10 - t;
h12 = h11 - t;
h13 = h12 - t;
h14 = h13 - t;
h15 = h14 - t;
h16 = h15 - t;
h17 = h16 - t;
h18 = h17 - t;
h19 = h18 - t;
h20 = h19 - t;

aa = a2a = a3a = a4a = a5a = a6a = a7a = a8a = a9a = a10a = a11a = a12a = a13a = a14a = a15a = a16a = a17a =
a18a = a19a = a20a = 0;
hh = h2h = h3h = h4h = h5h = h6h = h7h = h8h = h9h = h10h = h11h = h12h = h13h = h14h = h15h = h16h = h17h =
h18h = h19h = h20h = 0;

for (; b <= z; b++) {
	if (masiv[b] < x) {
		if (masiv[b] >= a10) {
			if (masiv[b] >= a5) {
				if (masiv[b] >= a3) {
					if (masiv[b] >= a) {
						aa++;
						continue;
					}
					if (masiv[b] >= a2)   a2a++;
					else   a3a++;
					continue;
				}
				else {
					if (masiv[b] >= a4)   a4a++;
					else   a5a++;
					continue;
				}
			}
```

```
        else {
                if (masiv[b] >= a8) {
                        if (masiv[b] >= a6) {
                                a6a++;
                                continue;
                        }
                        if (masiv[b] >= a7)   a7a++;
                        else   a8a++;
                        continue;
                }
                else {
                        if (masiv[b] >= a9)   a9a++;
                        else   a10a++;
                        continue;
                }
        }
}
else {
        if (masiv[b] >= a15) {
                if (masiv[b] >= a13) {
                        if (masiv[b] >= a11) {
                                a11a++;
                                continue;
                        }
                        if (masiv[b] >= a12) a12a++;
                        else   a13a++;
                        continue;
                }
                else {
                        if (masiv[b] >= a14)   a14a++;
                        else   a15a++;
                        continue;
                }
                }
                else {
                        if (masiv[b] >= a18) {
                                if (masiv[b] >= a16) {
                                        a16a++;
                                        continue;
                                }
                                if (masiv[b] >= a17)   a17a++;
                                else   a18a++;
                                continue;
                        }
                        else {
                                if (masiv[b] >= a19)   a19a++;
                                else   a20a++;
                                continue;
                        }
                }
        }
}
if (masiv[b] >= h20) {
        if (masiv[b] >= h10) {
                if (masiv[b] >= h5) {
                        if (masiv[b] >= h3) {
                                if (masiv[b] >= h) {
                                        hh++;
                                        continue;
                                }
                                if (masiv[b] >= h2)   h2h++;
                                else   h3h++;
                                continue;
                        }
                        else {
                                if (masiv[b] >= h4)   h4h++;
```

```
                        else   h5h++;
                        continue;
                }
        }
        else {
                if (masiv[b] >= h8) {
                        if (masiv[b] >= h6) {
                                h6h++;
                                continue;
                        }
                        if (masiv[b] >= h7)   h7h++;
                        else   h8h++;
                        continue;
                }
                else {
                        if (masiv[b] >= h9)   h9h++;
                        else   h10h++;
                        continue;
                }
        }
}
else {
        if (masiv[b] >= h15) {
                if (masiv[b] >= h13) {
                        if (masiv[b] >= h11) {
                                h11h++;
                                continue;
                        }
                        if (masiv[b] >= h12)   h12h++;
                        else   h13h++;
                        continue;
                }
                else {
                        if (masiv[b] >= h14)   h14h++;
                        else   h15h++;
                        continue;
                }
        }
        else {
                if (masiv[b] >= h18) {
                        if (masiv[b] >= h16) {
                                h16h++;
                                continue;
                        }
                        if (masiv[b] >= h17)   h17h++;
                        else   h18h++;
                        continue;
                }
                else {
                        if (masiv[b] >= h19)   h19h++;
                        else   h20h++;
                        continue;
                }
        }
    }
  }
}

as = v;
as20 = a20a = a20a + v;
a20h = a20a - 1;
as19 = a19a = a19a + a20a;
a19h = a19a - 1;
as18 = a18a = a18a + a19a;
a18h = a18a - 1;
as17 = a17a = a17a + a18a;
```

```
a17h = a17a - 1;
as16 = a16a = a16a + a17a;
a16h = a16a - 1;
as15 = a15a = a15a + a16a;
a15h = a15a - 1;
as14 = a14a = a14a + a15a;
a14h = a14a - 1;
as13 = a13a = a13a + a14a;
a13h = a13a - 1;
as12 = a12a = a12a + a13a;
a12h = a12a - 1;
as11 = a11a = a11a + a12a;
a11h = a11a - 1;
as10 = a10a = a10a + a11a;
a10h = a10a - 1;
as9 = a9a = a9a + a10a;
a9h = a9a - 1;
as8 = a8a = a8a + a9a;
a8h = a8a - 1;
as7 = a7a = a7a + a8a;
a7h = a7a - 1;
as6 = a6a = a6a + a7a;
a6h = a6a - 1;
as5 = a5a = a5a + a6a;
a5h = a5a - 1;
as4 = a4a = a4a + a5a;
a4h = a4a - 1;
as3 = a3a = a3a + a4a;
a3h = a3a - 1;
as2 = a2a = a2a + a3a;
a2h = a2a - 1;
aa = aa + a2a;
ah = aa - 1;
```

```
az20 = z;
az = hh = z - hh;
ha = hh + 1;
az2 = h2h = hh - h2h;
h2a = h2h + 1;
az3 = h3h = h2h - h3h;
h3a = h3h + 1;
az4 = h4h = h3h - h4h;
h4a = h4h + 1;
az5 = h5h = h4h - h5h;
h5a = h5h + 1;
az6 = h6h = h5h - h6h;
h6a = h6h + 1;
az7 = h7h = h6h - h7h;
h7a = h7h + 1;
az8 = h8h = h7h - h8h;
h8a = h8h + 1;
az9 = h9h = h8h - h9h;
h9a = h9h + 1;
az10 = h10h = h9h - h10h;
h10a = h10h + 1;
az11 = h11h = h10h - h11h;
h11a = h11h + 1;
az12 = h12h = h11h - h12h;
h12a = h12h + 1;
az13 = h13h = h12h - h13h;
h13a = h13h + 1;
az14 = h14h = h13h - h14h;
h14a = h14h + 1;
az15 = h15h = h14h - h15h;
h15a = h15h + 1;
az16 = h16h = h15h - h16h;
```

```
h16a = h16h + 1;
az17 = h17h = h16h - h17h;
h17a = h17h + 1;
az18 = h18h = h17h - h18h;
h18a = h18h + 1;
az19 = h19h = h18h - h19h;
h19a = h19h + 1;
h20h = h19h - h20h;
h20a = h20h + 1;

b = v;

for (; b <= z; ) {
        if (masiv[b] < x) {
                if (masiv[b] >= a10) {
                        if (masiv[b] >= a5) {
                                if (masiv[b] >= a3) {
                                        if (masiv[b] >= a) {
                                                if (b <= a2a) {
                                                        if (b >= as2) {
                                                                if (b < a2a) {
                                                                        b = a2a;
                                                                        continue;
                                                                }
                                                                if (b == ah) {
                                                                        b++;
                                                                        continue;
                                                                }
                                                                if (a2a < ah)   a2a++;
                                                                b = a2a;
                                                                continue;
                                                        }
                                                }
                                                vv = masiv[b];
                                                masiv[b] = masiv[a2a];
                                                masiv[a2a] = vv;
                                                if (b < a2a) {
                                                        if (a2a < ah) a2a++;
                                                        continue;
                                                }
                                                if (a2a < ah)   a2a++;
                                                b++;
                                                continue;
                                        }
                                        if (masiv[b] >= a2) {
                                                if (b <= a3a) {
                                                        if (b >= as3) {
                                                                if (b < a3a) {
                                                                        b = a3a;
                                                                        continue;
                                                                }
                                                                if (b == a2h) {
                                                                        b++;
                                                                        continue;
                                                                }
                                                                if (a3a < a2h)   a3a++;
                                                                b = a3a;
                                                                continue;
                                                        }
                                                }
                                                vv = masiv[b];
                                                masiv[b] = masiv[a3a];
                                                masiv[a3a] = vv;
                                                if (b < a3a) {
                                                        if (a3a < a2h) a3a++;
                                                        continue;
```

```
                        }
                        if (a3a < a2h)   a3a++;
                        b++;
                        continue;
                }
                else {
                        if (b <= a4a) {
                                if (b >= as4) {
                                        if (b < a4a) {
                                                b = a4a;
                                                continue;
                                        }
                                        if (b == a3h) {
                                                b++;
                                                continue;
                                        }
                                        if (a4a < a3h)   a4a++;
                                        b = a4a;
                                        continue;
                                }
                        }
                        vv = masiv[b];
                        masiv[b] = masiv[a4a];
                        masiv[a4a] = vv;
                        if (b < a4a) {
                                if (a4a < a3h) a4a++;
                                continue;
                        }
                        if (a4a < a3h)   a4a++;
                        b++;
                        continue;
                }
        }
        else {
                if (masiv[b] >= a4) {
                        if (b <= a5a) {
                                if (b >= as5) {
                                        if (b < a5a) {
                                                b = a5a;
                                                continue;
                                        }
                                        if (b == a4h) {
                                                b++;
                                                continue;
                                        }
                                        if (a5a < a4h)   a5a++;
                                        b = a5a;
                                        continue;
                                }
                        }
                        vv = masiv[b];
                        masiv[b] = masiv[a5a];
                        masiv[a5a] = vv;
                        if (b < a5a) {
                                if (a5a < a4h) a5a++;
                                continue;
                        }
                        if (a5a < a4h)   a5a++;
                        b++;
                        continue;
                }
                else {
                        if (b <= a6a) {
                                if (b >= as6) {
                                        if (b < a6a) {
                                                b = a6a;
```

```
                                        continue;
                                }
                                if (b == a5h) {
                                        b++;
                                        continue;
                                }
                                if (a6a < a5h)   a6a++;
                                b = a6a;
                                continue;
                        }
                }
                vv = masiv[b];
                masiv[b] = masiv[a6a];
                masiv[a6a] = vv;
                if (b < a6a) {
                        if (a6a < a5h) a6a++;
                        continue;
                }
                if (a6a < a5h)   a6a++;
                b++;
                continue;
        }
    }
}
else {
    if (masiv[b] >= a8) {
        if (masiv[b] >= a6) {
                if (b <= a7a) {
                        if (b >= as7) {
                                if (b < a7a) {
                                        b = a7a;
                                        continue;
                                }
                                if (b == a6h) {
                                        b++;
                                        continue;
                                }
                                if (a7a < a6h)   a7a++;
                                b = a7a;
                                continue;
                        }
                }
                vv = masiv[b];
                masiv[b] = masiv[a7a];
                masiv[a7a] = vv;
                if (b < a7a) {
                        if (a7a < a6h) a7a++;
                        continue;
                }
                if (a7a < a6h)   a7a++;
                b++;
                continue;
    }
    if (masiv[b] >= a7) {
            if (b <= a8a) {
                    if (b >= as8) {
                            if (b < a8a) {
                                    b = a8a;
                                    continue;
                            }
                            if (b == a7h) {
                                    b++;
                                    continue;
                            }
                            if (a8a < a7h)   a8a++;
                            b = a8a;
```

```
                        continue;
                }
        }
        vv = masiv[b];
        masiv[b] = masiv[a8a];
        masiv[a8a] = vv;
        if (b < a8a) {
                if (a8a < a7h) a8a++;
                continue;
        }
        if (a8a < a7h)   a8a++;
        b++;
        continue;
}
else {
        if (b <= a9a) {
                if (b >= as9) {
                        if (b < a9a) {
                                b = a9a;
                                continue;
                        }
                        if (b == a8h) {
                                b++;
                                continue;
                        }
                        if (a9a < a8h)   a9a++;
                        b = a9a;
                        continue;
                }
        }
        vv = masiv[b];
        masiv[b] = masiv[a9a];
        masiv[a9a] = vv;
        if (b < a9a) {
                if (a9a < a8h) a9a++;
                continue;
        }
        if (a9a < a8h)   a9a++;
        b++;
        continue;
}
else {
        if (masiv[b] >= a9) {
                if (b <= a10a) {
                        if (b >= as10) {
                                if (b < a10a) {
                                        b = a10a;
                                        continue;
                                }
                                if (b == a9h) {
                                        b++;
                                        continue;
                                }
                                if (a10a < a9h)   a10a++;
                                b = a10a;
                                continue;
                        }
                }
                vv = masiv[b];
                masiv[b] = masiv[a10a];
                masiv[a10a] = vv;
                if (b < a10a) {
                        if (a10a < a9h) a10a++;
                        continue;
                }
```

```
                        if (a10a < a9h)   a10a++;
                        b++;
                        continue;
                }
                else {
                        if (b <= a11a) {
                                if (b >= as11) {
                                        if (b < a11a) {
                                                b = a11a;
                                                continue;
                                        }
                                        if (b == a10h) {
                                                b++;
                                                continue;
                                        }
                                        if (a11a < a10h)   a11a++;
                                        b = a11a;
                                        continue;
                                }
                        }
                        vv = masiv[b];
                        masiv[b] = masiv[a11a];
                        masiv[a11a] = vv;
                        if (b < a11a) {
                                if (a11a < a10h) a11a++;
                                continue;
                        }
                        if (a11a < a10h)   a11a++;
                        b++;
                        continue;
                }
        }
}
else {
        if (masiv[b] >= a15) {
                if (masiv[b] >= a13) {
                        if (masiv[b] >= a11) {
                                if (b <= a12a) {
                                        if (b >= as12) {
                                                if (b < a12a) {
                                                        b = a12a;
                                                        continue;
                                                }
                                                if (b == a11h) {
                                                        b++;
                                                        continue;
                                                }
                                                if (a12a < a11h)   a12a++;
                                                b = a12a;
                                                continue;
                                        }
                                }
                                vv = masiv[b];
                                masiv[b] = masiv[a12a];
                                masiv[a12a] = vv;
                                if (b < a12a) {
                                        if (a12a < a11h) a12a++;
                                        continue;
                                }
                                if (a12a < a11h)   a12a++;
                                b++;
                                continue;
                        }
                        if (masiv[b] >= a12) {
                                if (b <= a13a) {
```

```
                        if (b >= as13) {
                                if (b < a13a) {
                                        b = a13a;
                                        continue;
                                }
                                if (b == a12h) {
                                        b++;
                                        continue;
                                }
                                if (a13a < a12h)   a13a++;
                                b = a13a;
                                continue;
                        }
                }
                vv = masiv[b];
                masiv[b] = masiv[a13a];
                masiv[a13a] = vv;
                if (b < a13a) {
                        if (a13a < a12h) a13a++;
                        continue;
                }
                if (a13a < a12h)   a13a++;
                b++;
                continue;
        }
        else {
                if (b <= a14a) {
                        if (b >= as14) {
                                if (b < a14a) {
                                        b = a14a;
                                        continue;
                                }
                                if (b == a13h) {
                                        b++;
                                        continue;
                                }
                                if (a14a < a13h)   a14a++;
                                b = a14a;
                                continue;
                        }
                }
                vv = masiv[b];
                masiv[b] = masiv[a14a];
                masiv[a14a] = vv;
                if (b < a14a) {
                        if (a14a < a13h) a14a++;
                        continue;
                }
                if (a14a < a13h)   a14a++;
                b++;
                continue;
        }
}
else {
        if (masiv[b] >= a14) {
                if (b <= a15a) {
                        if (b >= as15) {
                                if (b < a15a) {
                                        b = a15a;
                                        continue;
                                }
                                if (b == a14h) {
                                        b++;
                                        continue;
                                }
                                if (a15a < a14h)   a15a++;
```

```
                        b = a15a;
                        continue;
                }
        }
        vv = masiv[b];
        masiv[b] = masiv[a15a];
        masiv[a15a] = vv;
        if (b < a15a) {
                if (a15a < a14h) a15a++;
                continue;
        }
        if (a15a < a14h)   a15a++;
        b++;
        continue;
}
else {
        if (b <= a16a) {
                if (b >= as16) {
                        if (b < a16a) {
                                b = a16a;
                                continue;
                        }
                        if (b == a15h) {
                                b++;
                                continue;
                        }
                        if (a16a < a15h)   a16a++;
                        b = a16a;
                        continue;
                }
        }
        vv = masiv[b];
        masiv[b] = masiv[a16a];
        masiv[a16a] = vv;
        if (b < a16a) {
                if (a16a < a15h) a16a++;
                continue;
        }
        if (a16a < a15h)   a16a++;
        b++;
        continue;
                }
        }
}
else {
        if (masiv[b] >= a18) {
                if (masiv[b] >= a16) {
                        if (b <= a17a) {
                                if (b >= as17) {
                                        if (b < a17a) {
                                                b = a17a;
                                                continue;
                                        }
                                        if (b == a16h) {
                                                b++;
                                                continue;
                                        }
                                        if (a17a < a16h)   a17a++;
                                        b = a17a;
                                        continue;
                                }
                        }
                        vv = masiv[b];
                        masiv[b] = masiv[a17a];
                        masiv[a17a] = vv;
                        if (b < a17a) {
```

```
                        if (a17a < a16h) a17a++;
                        continue;
                }
                if (a17a < a16h)   a17a++;
                b++;
                continue;
        }
        if (masiv[b] >= a17) {
                if (b <= a18a) {
                        if (b >= as18) {
                                if (b < a18a) {
                                        b = a18a;
                                        continue;
                                }
                                if (b == a17h) {
                                        b++;
                                        continue;
                                }
                                if (a18a < a17h)   a18a++;
                                b = a18a;
                                continue;
                        }
                }
                vv = masiv[b];
                masiv[b] = masiv[a18a];
                masiv[a18a] = vv;
                if (b < a18a) {
                        if (a18a < a17h) a18a++;
                        continue;
                }
                if (a18a < a17h)   a18a++;
                b++;
                continue;
        }
        else {
                if (b <= a19a) {
                        if (b >= as19) {
                                if (b < a19a) {
                                        b = a19a;
                                        continue;
                                }
                                if (b == a18h) {
                                        b++;
                                        continue;
                                }
                                if (a19a < a18h)   a19a++;
                                b = a19a;
                                continue;
                        }
                }
                vv = masiv[b];
                masiv[b] = masiv[a19a];
                masiv[a19a] = vv;
                if (b < a19a) {
                        if (a19a < a18h) a19a++;
                        continue;
                }
                if (a19a < a18h)   a19a++;
                b++;
                continue;
        }
}
else {
        if (masiv[b] >= a19) {
                if (b <= a20a) {
                        if (b >= as20) {
```

```
                                        if (b < a20a) {
                                                b = a20a;
                                                continue;
                                        }
                                        if (b == a19h) {
                                                b++;
                                                continue;
                                        }
                                        if (a20a < a19h)   a20a++;
                                        b = a20a;
                                        continue;
                                }
                        }
                        vv = masiv[b];
                        masiv[b] = masiv[a20a];
                        masiv[a20a] = vv;
                        if (b < a20a) {
                                if (a20a < a19h) a20a++;
                                continue;
                        }
                        if (a20a < a19h)   a20a++;
                        b++;
                        continue;
                }
                else {
                        if (b <= v) {
                                if (b >= as) {
                                        if (b < v) {
                                                b = v;
                                                continue;
                                        }
                                        if (b == a20h) {
                                                b++;
                                                continue;
                                        }
                                        if (v < a20h)   v++;
                                        b = v;
                                        continue;
                                }
                                vv = masiv[b];
                                masiv[b] = masiv[v];
                                masiv[v] = vv;
                                if (b < v) {
                                        if (v < a20h) v++;
                                        continue;
                                }
                                if (v < a20h)   v++;
                                b++;
                                continue;
                        }
                }
        }
}
if (masiv[b] >= h20) {
        if (masiv[b] >= h10) {
                if (masiv[b] >= h5) {
                        if (masiv[b] >= h3) {
                                if (masiv[b] >= h) {
                                        if (b >= z) {
                                                if (b <= az20) break;

                                        }
                                        vv = masiv[b];
                                        masiv[b] = masiv[z];
```

```
                masiv[z] = vv;
                if (b < z) {
                        if (z > ha) z--;
                        continue;
                }
                if (z > ha)   z--;
                b++;
                continue;
        }
        if (masiv[b] >= h2) {
                if (b >= hh) {
                        if (b <= az) {
                                b = az;
                                b++;
                                continue;
                        }
                }
                vv = masiv[b];
                masiv[b] = masiv[hh];
                masiv[hh] = vv;
                if (b < hh) {
                        if (hh > h2a) hh--;
                        continue;
                }
                if (hh > h2a)   hh--;
                b++;
                continue;
        }
        else {
                if (b >= h2h) {
                        if (b <= az2) {
                                b = az2;
                                b++;
                                continue;
                        }
                }
                vv = masiv[b];
                masiv[b] = masiv[h2h];
                masiv[h2h] = vv;
                if (b < h2h) {
                        if (h2h > h3a) h2h--;
                        continue;
                }
                if (h2h > h3a)   h2h--;
                b++;
                continue;
        }
}
else {
        if (masiv[b] >= h4) {
                if (b >= h3h) {
                        if (b <= az3) {
                                b = az3;
                                b++;
                                continue;
                        }
                }
                vv = masiv[b];
                masiv[b] = masiv[h3h];
                masiv[h3h] = vv;
                if (b < h3h) {
                        if (h3h > h4a) h3h--;
                        continue;
                }
                if (h3h > h4a)   h3h--;
                b++;
```

```
                        continue;
                }
                else {
                        if (b >= h4h) {
                                if (b <= az4) {
                                        b = az4;
                                        b++;
                                        continue;
                                }
                        }
                        vv = masiv[b];
                        masiv[b] = masiv[h4h];
                        masiv[h4h] = vv;
                        if (b < h4h) {
                                if (h4h > h5a) h4h--;
                                continue;
                        }
                        if (h4h > h5a)   h4h--;
                        b++;
                        continue;
                }
        }
}
else {
        if (masiv[b] >= h8) {
                if (masiv[b] >= h6) {
                        if (b >= h5h) {
                                if (b <= az5) {
                                        b = az5;
                                        b++;
                                        continue;
                                }
                        }
                        vv = masiv[b];
                        masiv[b] = masiv[h5h];
                        masiv[h5h] = vv;
                        if (b < h5h) {
                                if (h5h > h6a) h5h--;
                                continue;
                        }
                        if (h5h > h6a)   h5h--;
                        b++;
                        continue;
                }
                if (masiv[b] >= h7) {
                        if (b >= h6h) {
                                if (b <= az6) {
                                        b = az6;
                                        b++;
                                        continue;
                                }
                        }
                        vv = masiv[b];
                        masiv[b] = masiv[h6h];
                        masiv[h6h] = vv;
                        if (b < h6h) {
                                if (h6h > h7a) h6h--;
                                continue;
                        }
                        if (h6h > h7a)   h6h--;
                        b++;
                        continue;
                }
                else {
                        if (b >= h7h) {
                                if (b <= az7) {
```

```
                                b = az7;
                                b++;
                                continue;
                        }
                }
                vv = masiv[b];
                masiv[b] = masiv[h7h];
                masiv[h7h] = vv;
                if (b < h7h) {
                        if (h7h > h8a) h7h--;
                        continue;
                }
                if (h7h > h8a)   h7h--;
                b++;
                continue;
        }
}
else {
        if (masiv[b] >= h9) {
                if (b >= h8h) {
                        if (b <= az8) {
                                b = az8;
                                b++;
                                continue;
                        }
                }
                vv = masiv[b];
                masiv[b] = masiv[h8h];
                masiv[h8h] = vv;
                if (b < h8h) {
                        if (h8h > h9a) h8h--;
                        continue;
                }
                if (h8h > h9a)   h8h--;
                b++;
                continue;
        }
        else {
                if (b >= h9h) {
                        if (b <= az9) {
                                b = az9;
                                b++;
                                continue;
                        }
                }
                vv = masiv[b];
                masiv[b] = masiv[h9h];
                masiv[h9h] = vv;
                if (b < h9h) {
                        if (h9h > h10a) h9h--;
                        continue;
                }
                if (h9h > h10a)   h9h--;
                b++;
                continue;
        }
    }
}
else {
        if (masiv[b] >= h15) {
                if (masiv[b] >= h13) {
                        if (masiv[b] >= h11) {
                                if (b >= h10h) {
                                        if (b <= az10) {
                                                b = az10;
```

```
                        b++;
                        continue;
                }
        }
        vv = masiv[b];
        masiv[b] = masiv[h10h];
        masiv[h10h] = vv;
        if (b < h10h) {
                if (h10h > h11a) h10h--;
                continue;
        }
        if (h10h > h11a)   h10h--;
        b++;
        continue;
}
if (masiv[b] >= h12) {
        if (b >= h11h) {
                if (b <= az11) {
                        b = az11;
                        b++;
                        continue;
                }
        }
        vv = masiv[b];
        masiv[b] = masiv[h11h];
        masiv[h11h] = vv;
        if (b < h11h) {
                if (h11h > h12a) h11h--;
                continue;
        }
        if (h11h > h12a)   h11h--;
        b++;
        continue;
        }
        else {
                if (b >= h12h) {
                        if (b <= az12) {
                                b = az12;
                                b++;
                                continue;
                        }
                }
                vv = masiv[b];
                masiv[b] = masiv[h12h];
                masiv[h12h] = vv;
                if (b < h12h) {
                        if (h12h > h13a) h12h--;
                        continue;
                }
                if (h12h > h13a)   h12h--;
                b++;
                continue;
        }
}
else {
        if (masiv[b] >= h14) {
                if (b >= h13h) {
                        if (b <= az13) {
                                b = az13;
                                b++;
                                continue;
                        }
                }
                vv = masiv[b];
                masiv[b] = masiv[h13h];
                masiv[h13h] = vv;
```

```
                        if (b < h13h) {
                                if (h13h > h14a) h13h--;
                                continue;
                        }
                        if (h13h > h14a)   h13h--;
                        b++;
                        continue;
                }
                else {
                        if (b >= h14h) {
                                if (b <= az14) {
                                        b = az14;
                                        b++;
                                        continue;
                                }
                        }
                        vv = masiv[b];
                        masiv[b] = masiv[h14h];
                        masiv[h14h] = vv;
                        if (b < h14h) {
                                if (h14h > h15a) h14h--;
                                continue;
                        }
                        if (h14h > h15a)   h14h--;
                        b++;
                        continue;
                }
        }
}
else {
        if (masiv[b] >= h18) {
                if (masiv[b] >= h16) {
                        if (b >= h15h) {
                                if (b <= az15) {
                                        b = az15;
                                        b++;
                                        continue;
                                }
                        }
                        vv = masiv[b];
                        masiv[b] = masiv[h15h];
                        masiv[h15h] = vv;
                        if (b < h15h) {
                                if (h15h > h16a) h15h--;
                                continue;
                        }
                        if (h15h > h16a)   h15h--;
                        b++;
                        continue;
                }
                if (masiv[b] >= h17) {
                        if (b >= h16h) {
                                if (b <= az16) {
                                        b = az16;
                                        b++;
                                        continue;
                                }
                        }
                        vv = masiv[b];
                        masiv[b] = masiv[h16h];
                        masiv[h16h] = vv;
                        if (b < h16h) {
                                if (h16h > h17a) h16h--;
                                continue;
                        }
                        if (h16h > h17a)   h16h--;
```

```
                b++;
                continue;
        }
        else {
                if (b >= h17h) {
                        if (b <= az17) {
                                b = az17;
                                b++;
                                continue;
                        }
                }
                vv = masiv[b];
                masiv[b] = masiv[h17h];
                masiv[h17h] = vv;
                if (b < h17h) {
                        if (h17h > h18a) h17h--;
                        continue;
                }
                if (h17h > h18a)   h17h--;
                b++;
                continue;
        }
}
else {
        if (masiv[b] >= h19) {
                if (b >= h18h) {
                        if (b <= az18) {
                                b = az18;
                                b++;
                                continue;
                        }
                }
                vv = masiv[b];
                masiv[b] = masiv[h18h];
                masiv[h18h] = vv;
                if (b < h18h) {
                        if (h18h > h19a) h18h--;
                        continue;
                }
                if (h18h > h19a)   h18h--;
                b++;
                continue;
        }
        else {
                if (b >= h19h) {
                        if (b <= az19) {
                                b = az19;
                                b++;
                                continue;
                        }
                }
                vv = masiv[b];
                masiv[b] = masiv[h19h];
                masiv[h19h] = vv;
                if (b < h19h) {
                        if (h19h > h20a) h19h--;
                        continue;
                }
                if (h19h > h20a)   h19h--;
                b++;
                continue;
        }
                }
        }
}
```

```
                b++;
        }

        b = v = aa;
        z = h20h;
        a20 = x;
        h = h20;
}

// Приступаем к сортировке массива (2 этап).

aa = an;
x = y;
b = bb;
z = 0;
for (d = 0; d <= b; d++) {
        v = vv = z;
        aa = aa + t;
        for (; v < s; v++) {
                if (masiv[v] < aa) {
                        if (v == x) goto mit;
                        if (masiv[v + 1] >= aa) {
                        mit:
                                if (vv == v) {
                                        z = ++v;
                                        break;
                                }
                                z = v = ++v;
                                for (; v > vv; v--) {
                                        h = masiv[vv];
                                        for (c = (vv + 1); c < v; c++) if (h < masiv[c]) h = masiv[c];
                                        for (c = vv; c < v; c++) if (h == masiv[c]) y = c;
                                        if (masiv[y] == masiv[v - 1]) continue;
                                        masiv[y] = masiv[v - 1];
                                        masiv[v - 1] = h;
                                }
                                break;
                        }
                        continue;
                }
                if (masiv[v] >= aa) break;
        }
}

// для предотвращения закрытия окна, я применяю
// вот этот "кусочек":  -  cin >> i;

        cout << "Массив отсортирован.\n";
        cout << "Чтобы вывести массив, введите число и нажмите Enter\n";
        cin >> i;
        for (a = 0; a < s; a++) cout << masiv[a] << ' ';
        cin >> i;
        cout << i;
        return 0;
}
```

Конец алгоритма.