Министерство транспорта РФ

Федеральное государственное автономное

образовательное учреждение высшего образования

«Российский университет транспорта» (МИИТ)

Лабораторная работа №8

«Пирамидальная сортировка»

по дисциплине

«Методы программирования»

Выполнил:

студент группы ТКИ-311,

Гаврилин-Кирьянов И.А.

Проверил:

к.т.н., доц. Логинова Л. Н,

к.т.н., доц. Сафронов А. И.

Москва - 2022 г.

**Цель работы:**

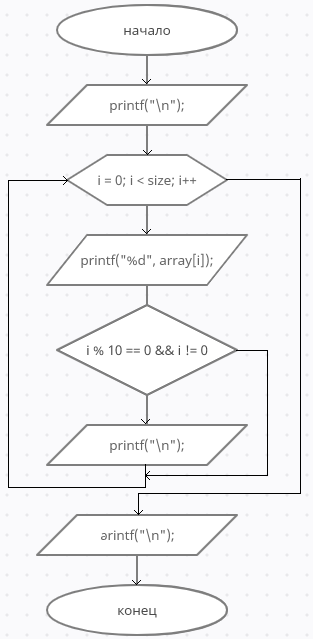
Изучение алгоритма пирамидальной сортировки.

**Задача:**

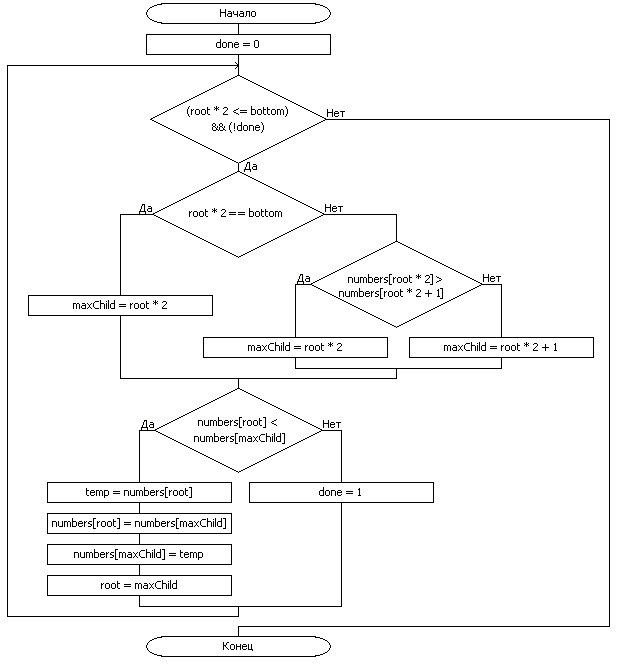
Реализовать алгоритм пирамидальной сортировки на языке С

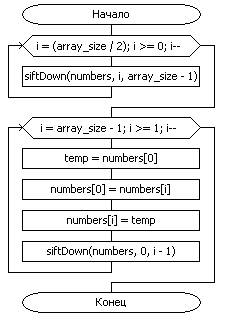
**Блок-схемы:**

Функция вывода массива:



Функция пирамидальной сортировки:





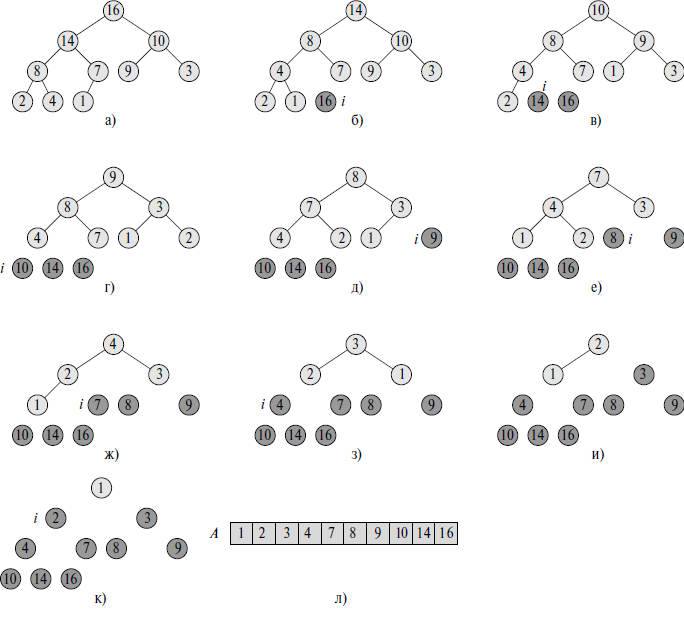
**Тестовые примеры:**

Массивы длиной 10, 1000, 10000 и 100000 элементов.

Имеем массив:

16, 14, 10, 8, 7, 9, 3, 2, 4, 1

Проверим алгоритм вручную:



Сортировка завершена.

Данный алгоритм выстраивает элементы в виде сортирующего дерева, а затем удаляет элементы из корня по одному за раз и перестраивает дерево. Процесс продолжается, пока не останется один элемент.

**Код программы:**

#include <stdio.h>

#include <stdlib.h>

int n;

void siftDown(int \*numbers, int root, int bottom)

{

int maxChild;

int done = 0;

while ((root \* 2 <= bottom) && (!done))

{

if (root \* 2 == bottom)

maxChild = root \* 2;

else if (numbers[root \* 2] > numbers[root \* 2 + 1])

maxChild = root \* 2;

else

maxChild = root \* 2 + 1;

if (numbers[root] < numbers[maxChild])

{

int temp = numbers[root];

numbers[root] = numbers[maxChild];

numbers[maxChild] = temp;

root = maxChild;

}

else

done = 1;

}

}

void heapSort(int \*numbers, int array\_size)

{

for (int i = (array\_size / 2); i >= 0; i--)

siftDown(numbers, i, array\_size - 1);

for (int i = array\_size - 1; i >= 1; i--)

{

int temp = numbers[0];

numbers[0] = numbers[i];

numbers[i] = temp;

siftDown(numbers, 0, i - 1);

}

}

void print(int array[n], int n)

{

printf("\n");

for (int i = 0; i < n; i++)

{

printf("%d ", array[i]);

if (i % 15 == 14)

{

printf("\n");

}

}

printf("\n");

return;

}

int main()

{

printf("Enter n size: ");

scanf("%d", &n);

printf("%d", n);

int array[n];

for (int i = 0; i < n; i++)

scanf("%d", &array[i]);

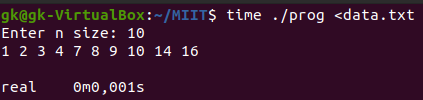
heapSort(array, n);

print(array, n);

}

**Результаты программы:**

10 элементов:

****

Программа выполняется за 0,001 секунды

1000 элементов:

Изображение выглядит как текст

Автоматически созданное описание

10000 элементов:

Изображение выглядит как текст

Автоматически созданное описание

100000 элементов:

Изображение выглядит как текст

Автоматически созданное описание

**Вывод:**

Я изучил и реализовал алгоритм пирамидальной сортировки. Она похожа на сортировку выбором, где мы сначала ищем максимальный элемент и помещаем его в конец. Далее мы повторяем ту же операцию для оставшихся элементов. Пирамидальная сортировка не использует дополнительной памяти. Данная сортировка на почти отсортированных массивах работает также долго, выигрыш ее получается только на больших n. Стоит заметить, что скорость пирамидальной сортировки не изменится, если будем работать уже с отсортированным массивом