МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Вятский государственный университет»

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

Отчет

Лабораторная работа №5 по дисциплине

«Программирование»

Исследование алгоритмов сортировки

Выполнил студент группы ПИб-1301 \_\_\_\_\_\_\_\_\_\_/Суров К. С./

Проверил преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Чистяков Г.А./

Киров 2019

1. Цель работы: получить базовые сведения о наиболее известных алгоритмах сортировки, изучить принципы работы с текстовыми файлами.
2. Задание(Вариант 810):
3. Реализовать сортировку данных с помощью «пузырькового» алгоритма.
4. Реализовать сортировку данных с помощью пирамидального алгоритма.
5. В обоих случаях необходимо предусмотреть возможность изменения компаратора (реализация компаратора в виде передаваемой в подпрограмму функции).
6. Считывание и вывод данных необходимо производить из текстового файла.
7. Для демонстрации работы программных реализаций самостоятельно подготовить варианты входных данных (при этом объём текстовых файлов должен позволять оценить скорость работы программ).
8. Разработка алгоритма(схемы)
9. Код программ:
10. Сортировка «пузырьком» (возрастание):

Program boobleV;

Type cm=function(o,t:longint):boolean;

Var a:array of longint;

n,i:longint;

inp,outp:text;

Function comp(o,t:longint):boolean;

begin

If o>t then comp:=true

else comp:=false;

end;

Procedure sort(c:cm);

Var count,i,l:longint;

Begin

count:=0;

While count<n do begin

i:=0;

For i:=0 to (n-2) do begin

If c(a[i],a[i+1])=true then begin

l:=a[i];

a[i]:=a[i+1];

a[i+1]:=l;

end;

end;

count:=count+1;

end;

end;

Begin

assign(inp,'input.txt');

Reset(inp);

Readln(inp,n);

SetLength(a,n);

i:=0;

For i:=0 to (n-1) do

Read(inp,a[i]);

Close(inp);

sort(@comp);

assign(outp,'output.txt');

rewrite(outp);

i:=0;

For i:=0 to (n-1) do

write(outp,a[i],' ');

Close(outp);

end.

1. Сортировка «пузырьком» (убывание):

Program boobleU;

Type cm=function(o,t:longint):boolean;

Var a:array of longint;

n,i:longint;

inp,outp:text;

Function comp(o,t:longint):boolean;

begin

If o<t then comp:=true

else comp:=false;

end;

Procedure sort(c:cm);

Var count,i,l:longint;

Begin

count:=0;

While count<n do begin

i:=0;

For i:=0 to (n-2) do begin

If c(a[i],a[i+1])=true then begin

l:=a[i];

a[i]:=a[i+1];

a[i+1]:=l;

end;

end;

count:=count+1;

end;

end;

Begin

assign(inp,'input.txt');

Reset(inp);

Readln(inp,n);

SetLength(a,n);

i:=0;

For i:=0 to (n-1) do

Read(inp,a[i]);

Close(inp);

sort(@comp);

assign(outp,'output.txt');

rewrite(outp);

i:=0;

For i:=0 to (n-1) do

write(outp,a[i],' ');

Close(outp);

end.

1. Пирамидальная сортировка(возрастание):

Program PirV;

Type cm=function(o,t:longint):boolean;

Var i,n,k,l:longint;

derevo:array of longint;

inp,outp:text;

Function comp(o,t:longint):boolean;

begin

If o>t then comp:=true

else comp:=false;

end;

Procedure sort(c:cm);

Var i,p,k,max,m,l:longint;

Begin

i:=n-1;

While i>1 do begin

l:=derevo[i];

derevo[i]:=derevo[1];

derevo[1]:=l;

i:=i-1;

m:=1; p:=1;

While (2\*p<=i) and (m=1) do begin

max:=2\*p; m:=0;

If (p\*2+1<=i) and (c(derevo[2\*p+1],derevo[max])=true) then max:=p\*2+1;

If c(derevo[p],derevo[max])=false then begin

l:=derevo[max];

derevo[max]:=derevo[p];

derevo[p]:=l;

m:=1;

p:=max;

end;

end;

end;

end;

Begin

assign(inp,'input.txt');

assign(outp,'output.txt');

Rewrite(outp);

Reset(inp);

Readln(inp,n);

n:=n+1;

SetLength(derevo,n);

i:=1;

While i<n do begin

Read(inp,derevo[i]);

k:=i;

While (i>1) and (comp(derevo[i],derevo[i div 2])=true) do begin

l:=derevo[i];

derevo[i]:=derevo[i div 2];

derevo[i div 2]:=l;

i:=i div 2;

end;

i:=k+1;

end;

sort(@comp);

Close(inp);

i:=1;

For i:=1 to (n-1) do

Write(outp,derevo[i],' ');

Close(outp);

end.

1. Пирамидальная сортировка(убывание):

Program PirU;

Type cm=function(o,t:longint):boolean;

Var i,n,k,l:longint;

derevo:array of longint;

inp,outp:text;

Function comp(o,t:longint):boolean;

begin

If o<t then comp:=true

else comp:=false;

end;

Procedure sort(c:cm);

Var i,p,k,max,m,l:longint;

Begin

i:=n-1;

While i>1 do begin

l:=derevo[i];

derevo[i]:=derevo[1];

derevo[1]:=l;

i:=i-1;

m:=1; p:=1;

While (2\*p<=i) and (m=1) do begin

max:=2\*p; m:=0;

If (p\*2+1<=i) and (c(derevo[2\*p+1],derevo[max])=true) then max:=p\*2+1;

If c(derevo[p],derevo[max])=false then begin

l:=derevo[max];

derevo[max]:=derevo[p];

derevo[p]:=l;

m:=1;

p:=max;

end;

end;

end;

end;

Begin

assign(inp,'input.txt');

assign(outp,'output.txt');

Rewrite(outp);

Reset(inp);

Readln(inp,n);

n:=n+1;

SetLength(derevo,n);

i:=1;

While i<n do begin

Read(inp,derevo[i]);

k:=i;

While (i>1) and (comp(derevo[i],derevo[i div 2])=true) do begin

l:=derevo[i];

derevo[i]:=derevo[i div 2];

derevo[i div 2]:=l;

i:=i div 2;

end;

i:=k+1;

end;

sort(@comp);

Close(inp);

i:=1;

For i:=1 to (n-1) do

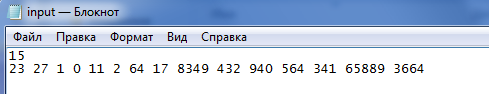
Write(outp,derevo[i],' ');

Close(outp);

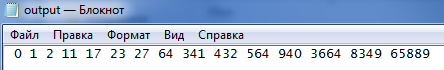
end.

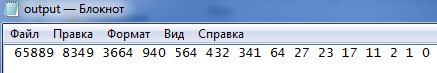
1. Результат работы:
2. Сортировка «пузырьком»:

Исходные данные:



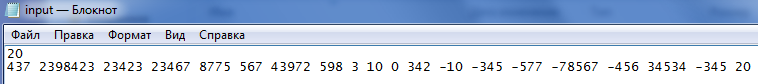
Результат:



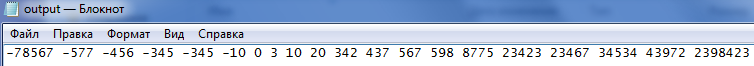


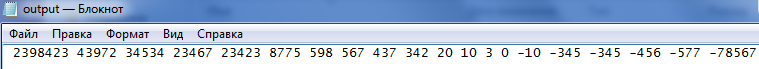
1. Пирамидальная сортировка:

Исходные данные:



Результат:





1. Выводы:

В ходе данной работы была реализована сортировка данных с помощью двух алгоритмов: «пузырькового» и пирамидального. Были получены навыки работы с текстовыми файлами.

Сложность написания программы была в передаче функции в процедуру, но данная проблема была решена путём создания нового типа.

Схемы:

1. Компаратор:
2. По возрастанию:

Изображение выглядит как текст, карта

Автоматически созданное описание

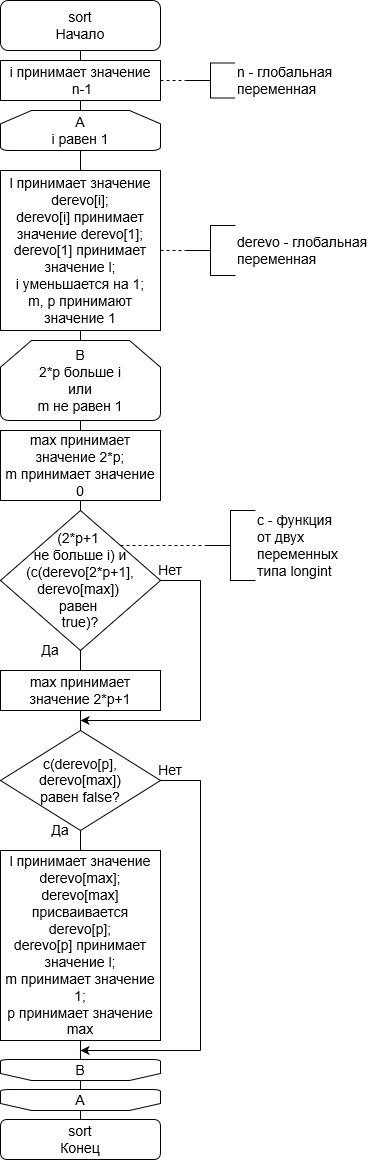
1. Изображение выглядит как текст, карта

   Автоматически созданное описаниеПо убыванию:
2. Процедуры сортировок:
3. «Пузырьковый» алгоритм:

Изображение выглядит как текст

Автоматически созданное описание

1. Пирамидальный алгоритм:



1. Основная программа:
2. «Пузырьковый» алгоритм:

Изображение выглядит как текст, карта

Автоматически созданное описание

1. Пирамидальный алгоритм:

