МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний авіаційний університет

Кафедра прикладної інформатики

Лабараторна робота №11

З дисципліни Теорія алгоритмів

Виконав:

Студент 1 курсу ІКІТ

Групи УС-112

Лазоренко Максим Олександрович

Перевірив : Ходаков Д.В

Київ 2016

**Двійкове** (або **Бінарне**) **дéрево пóшуку**([англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) *binary search tree*, BST) в [інформатиці](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0" \o "Інформатика) — [двійкове дерево](https://uk.wikipedia.org/wiki/%D0%94%D0%B2%D1%96%D0%B9%D0%BA%D0%BE%D0%B2%D0%B5_%D0%B4%D0%B5%D1%80%D0%B5%D0%B2%D0%BE" \o "Двійкове дерево), в якому кожній вершині *x* зіставлене певне значення val[*x*]. При цьому такі значення повинні задовольняти *умові впорядкованості*:[*[Джерело?](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%BA%D1%96%D0%BF%D0%B5%D0%B4%D1%96%D1%8F:%D0%9F%D0%BE%D1%81%D0%B8%D0%BB%D0%B0%D0%BD%D0%BD%D1%8F_%D0%BD%D0%B0_%D0%B4%D0%B6%D0%B5%D1%80%D0%B5%D0%BB%D0%B0" \o "Вікіпедія:Посилання на джерела)*]

* нехай *x* — довільна вершина двійкового дерева пошуку. Якщо вершина *y*знаходиться в лівому піддереві вершини *x*, то val[*y*] ≤ val[*x*].
* Якщо *у* знаходиться у правому піддереві *x*, то val[*y*] ≥ val[*x*].

Таке структурування дозволяє надрукувати усі значення у зростаючому порядку за допомогою простого алгоритму [центрованого обходу дерева](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%D1%85%D1%96%D0%B4_%D0%B4%D0%B5%D1%80%D0%B5%D0%B2%D0%B0" \o "Обхід дерева).[*[Джерело?](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%BA%D1%96%D0%BF%D0%B5%D0%B4%D1%96%D1%8F:%D0%9F%D0%BE%D1%81%D0%B8%D0%BB%D0%B0%D0%BD%D0%BD%D1%8F_%D0%BD%D0%B0_%D0%B4%D0%B6%D0%B5%D1%80%D0%B5%D0%BB%D0%B0" \o "Вікіпедія:Посилання на джерела)*]

Представляється таке дерево вузлами наступного вигляду:

\*Node = (element, key, left, right, parent). Доступ до дерева *T* здійснюється за допомогою посилання *root*.

Бінарні дерева пошуку набагато ефективніші в операціях пошуку, аніж лінійні структури, в яких витрати часу на пошук пропорційні [*O(n)*](https://uk.wikipedia.org/wiki/%D0%9D%D0%BE%D1%82%D0%B0%D1%86%D1%96%D1%8F_%D0%9B%D0%B0%D0%BD%D0%B4%D0%B0%D1%83), де n — розмір [масиву](https://uk.wikipedia.org/wiki/%D0%9C%D0%B0%D1%81%D0%B8%D0%B2_(%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85)" \o "Масив (структура даних))даних, тоді як в повному бінарному дереві цей час пропорційний в середньому *O(log2n)* або *O(h)*, де h — висота дерева (хоча гарантувати, що h не перевищує log2n можна лише для [збалансованих дерев](https://uk.wikipedia.org/wiki/%D0%97%D0%B1%D0%B0%D0%BB%D0%B0%D0%BD%D1%81%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D0%B4%D0%B5%D1%80%D0%B5%D0%B2%D0%BE" \o "Збалансоване дерево), які є ефективнішими на пошукових алгоритмах, аніж прості бінарні дерева пошуку).[*Джерело?*



**Код програми на с++**

#include<graphics.h>

#include<iostream>

#include<dos.h>

using namespace std;

void kolo1 (int X,int Y,int R);

void kolo2 (int X,int Y,int R);

int main(){

int gdriver=DETECT, gmode, errorcode;

initgraph(&gdriver,&gmode,"C\\TC\\BGI");

errorcode=graphresult();

if(errorcode!=grOk){

cout<<"Error"<<grapherrormsg(errorcode);

getch();

exit(1);} //Запуск графічоного режиму

int y=getmaxy()/2;

int x=getmaxx()/2;// кординати найбільшого кола по центру екрану

int r;

cin>>r; //введення радіусу кола

setcolor(15);

circle(x, y, r); //Малюємо перше найбільше коло

kolo1(x,y,r);

kolo2(x,y,r); //викликаємо функції які малюютьменші круги та рекурсивно створюють дерева

getch();

closegraph();

return 0;

}

void kolo1 (int X,int Y,int R){

int r=R;

R=3\*R/5;

circle(X+r,Y,R);

X+=r;

if(R>20){

kolo1(X,Y,R);

kolo2(X,Y,R);//знову викликаємо функції}}

void kolo2 (int X,int Y,int R){

int r=R;

R=3\*R/5;

circle(X-r,Y,R);

X-=r; if(R>20){

kolo1(X,Y,R);

kolo2(X,Y,R);//знову викликаємо функції}}

**Скріншот виконання програми:**

