3 OGP in C++

Oefening 120

Gegeven onderstaand hoofdprogramma, opgedeeld in drie stukken. Schrijf de klasse `Breuk` die hiervoor nodig is. (Haal de code uit het LaTeX-bestand dat je op Minerva vindt; dan zijn de kantlijnen netjes.) Enkele tips:

1. De (externe) functie `mijn_ggd` geeft de grootste gemene deler van twee gehele getallen.
2. Een breuk staat in zijn meest vereenvoudigde vorm als teller en noemer niet meer onderling deelbaar zijn. (Zijn ze dat nog wel, dan deel je teller en noemer door hun grootste gemene deler.) Bewaar een breuk steeds in zijn vereenvoudigde vorm! (Schrijf een (hulp)lidfunctie `normaliseer()`.)
3. Een stambreuk is een breuk waarvan de teller gelijk is aan 1.

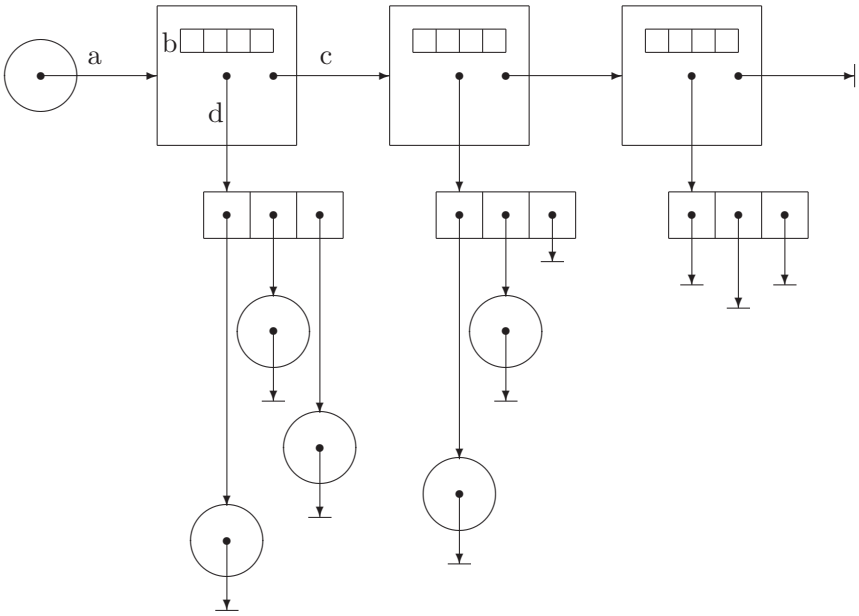
```
#include <iostream>
#include <fstream>
#include <string>  // nodig bij het inlezen van een breuk
#include <sstream> // vergelijk met Scanner(String) in Java
#include <cmath>   // abs berekent absolute waarde van een int (fabs is voor double)
#include <set>
using namespace std;

// zet deze functie in het bestand 'breuk.cpp'
int mijn_ggd(int a, int b){
    if(a < 0 || b < 0){
        return mijn_ggd(abs(a),abs(b));
    }
    if(a < b){
        return mijn_ggd(b,a);
    }
    if(b == 0){
        return a;
    }
    return mijn_ggd(b,a%b);
}

void deel1(){
    Breuk a(2,5);
    Breuk b(1,-2);

    cout << a << " + " << b << " = ";
    cout << (a+b) << " [-1/10 ?]" << endl;
    cout << "De tegengestelde breuk van " << a << " is " << -a << " [-2/5 ?]." << endl;

    Breuk f = a + b;
    cout << "De som van " << a << " en " << b << " is " << f << " [-1/10 ?]" << endl;
    Breuk g(f);
    cout << "en dat is gelijk aan de breuk " << g <<" [-1/10 ?]."<<endl;

    cout << a << " - " << b << " = ";
    cout << (a-b) << " [9/10 ?]" << endl;
    cout << a << " += " << b << " geeft als resultaat dat de breuk " << a << " verandert in ";
    a += b;
    cout << a << " [-1/10 ?]" << endl;
    cout << a << " -= " << b << " geeft als resultaat dat de breuk " << a << " verandert in ";
    a -= b;
```

```
    cout << a << " [2/5 ?]" << endl;

    cout << "Ik verhoog nu de breuk a=" << a <<" met 2 eenheden; dan is a=";
    cout << ++(++a) << " [12/5 ?]" << endl;

    cout << "Na dit uitschrijven zal b=" << b++ <<" ook met een eenheid verhoogd zijn, nl. ";
    cout << "b=" << b << " [1/2 ?]" << endl;

    Breuk c(2,3);
    Breuk d(3,4);
    Breuk e(1,2);
    (c -= d) += e;
    cout << "Indien hier 5/12 staat, heb je de operatoren -= en += goed geschreven: " << c << endl;
}

void deel2(){
    Breuk d(2,10);
    Breuk e(3);

    cout << d << " is stambreuk: "<<is_stambreuk(d)<<endl;

    Breuk f(3,4);
    cout << endl << "We starten van een breuk, en tellen er telkens een eenheid bij op: " << endl << endl;
    for(int i=0; i<10; i++){
        cout <<i<<" meer dan "<< f << " is " << (i+f) << " = " << (f+i) << endl;
    }

    cout << endl << "Al deze breuken in een verzameling: " << endl;
    set<Breuk> verz;
    for(int i=0; i<10; i++){
        verz.insert(i+f);
        verz.insert(f+i);
    }
    for(Breuk b : verz){
        cout << b << endl;
    }
}

void deel3(){
    Breuk a(13,12);
    Breuk b(2);
    Breuk c;
    cout<<endl<<"Geef c op (onder de vorm teller/noemer (bvb 13/12) of teller (bvb 13) : ";
    cin >> c;
    if(a == c){
        cout << a << " is gelijk aan " << c <<endl;
    }
    if(a != c){
        cout << a << " is niet gelijk aan " << c <<endl;
    }
}

int main(){
    deel1();
    deel2();
    deel3();
    return 0;
}
```

Oefening 121

Gegeven: een schets met bijhorende legende.

1. een pijl stelt een *raw* pointer voor; de geheugenplaats waar hij bewaard wordt is het zwarte bolletje van waaruit de pijl vertrekt.
2. een pijl die aankomt op een dwars streepje, stelt een nullpointer voor.
3. een grote cirkel is een object van de klasse `Schijf`.
4. een groot vierkant is een object van de klasse `Doos`.
5. een rechthoek onderverdeeld in 4 vierkantjes is een vector van type `T` en lengte 4.
6. een rechthoek onderverdeeld in 3 vierkantjes is een array van lengte 3. Deze arrays zijn dynamisch aangemaakt, aan de hand van de pointer die naar deze array wijst.



Gevraagd: Geef eerst voor de klassen `Schijf` en `Doos` de instantievariabelen. Implementeer daarna voor beide klassen de defaultconstructor, de destructor, de copyconstructor en de toekenningsoperator.
Voor de klasse `Doos` moet je nog het volgende weten: de defaultconstructor zal de vector `b` onmiddellijk grootte 4 geven. De pointer met naam `c` wordt de **nullpointer**. De pointer met naam `d` wijst naar een nieuwe array, die enkel de **nullpointer** bevat.