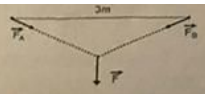

2. Eine 5m lange Kette ohne Eigengewicht ist an 3m entfernten Punkten aufgehängt. In ihrer Mitte hängt ein Gewicht F = 10 N. Berechnen Sie die Beträge der Zugkräfte im Seil.

3. An eine 4m lange Kette, deren Enden wieder 3m auseinander befestigt sind, hängt das Gewicht F = 10 N an einer Stelle, an der die Kette im Winkel von 90° geknickt wird. Berechnen Sie die Seilkräfte.

4. Suchen Sie die Punkte, die die Strecke zwischen $P_1 = (7; 3; 2)$ und $P_2 = (-1; 1; 3)$ dritteln.

1. Allgemeines Verständnis

Nr. 1	Aufgabe	Punkte
a)	Wie oft wird die folgende Schleife durchlaufen, d.h. welchen Wert liefert der Aufruf cont.size()?	2

```
vector<int> cont;
for (int i=1; i < 35; ++i)
{
      cont.push_back(17);
}
datei << cont.size() << " runs\n";
```

b) Wie oft wird die folgende Schleife durchlaufen, d.h. welchen Wert liefert der Aufruf cont.size() ? 2

```
list <int> cont;
for (int i=1; i < 35; i++) {
cont.push_back(17);
}
datei << cont.size() << " runs\n";
```

c) Wie oft wird die folgende Schleife durchlaufen, d.h. welchen Wert hat die Variable **summe** nach der Schleife? 2

```
int summe = 0;
for (int i=16; i > 1; i--) {
summe++;
}
datei << summe << " runs\n";
```

d) Wie oft wird die folgende Schleife durchlaufen, d.h. welchen Wert hat die Variable **summe** nach der Schleife? 2

```
int summe = 0;
for (int i=72; i > 12; i = i - 12) {
summe++;
}
```

e) Wie oft wird die folgende Schleife durchlaufen, d.h. welchen Wert hat die Variable **summe** nach der Schleife? 2

```
int summe = 0;
int i=23;
while (i<58) {
summe++;
++i;
}
```

f) Wie oft wird die folgende Schleife durchlaufen, d.h. welchen Wert hat die Variable **summe** nach der Schleife? 2

```
int summe = 0;
int i=12;
while (i<112) {
summe++;
i++;
}
```

4. Suchen Sie die Punkte, die die Strecke zwischen $P_1 = (7; 3; 2)$ und $P_2 = (-1; 1; 3)$ dritteln.

2. Begriffserklärung

Punkte

Nr. 2 **Hinweis 1**

person.h Die Headerdatei zu Klasse Person

```
#include <iostream>
#include <string>
using namespace std;

class Person {
public:
      Person(string n);
      virtual ~Person();
      void Setze(string n);
      virtual void Print() const;
private:
      string name;
};

ostream& operator<<(ostream& s, const Person& p);
```

person.cpp Die Codedatei zu Klasse Person

```
#include "Person.h"
#include <fstream>
using namespace std;

extern ofstream datei;

Person::Person(string n): name(n) {
      datei << "+P." << name << endl;
}
Person::~Person() {
      datei << "-P." << name << endl;
}
void Person::Setze(string n) { name=n; }
void Person::Print() const{
      datei << name;
}
ostream& operator<<(ostream& s, const Person& p) {
      p.Print();
      return s;
}
```

befestigt sind, hängt das Gewicht F = 10 N an d
90° geknickt wird. Berechnen Sie die Seilkräfte.

4. Suchen Sie die Punkte, die die Strecke zwischen $P_1 = (7; 3; 2)$ und $P_2 = (-1; 1; 3)$ dritteln.

Nr. 2 **Hinweis 2** Punkte

mann.h

Die Headerdatei zu Klasse Mann

```
#include "Person.h"
#include <iostream>
using namespace std;

class Mann : public Person {
      int age;
public:
      Mann(string n, int w);
      virtual ~Mann();
      Mann(const Mann& m2);
      void Setze(string w);
      virtual void Print() const;
};

ostream& operator<<(ostream& s, const Mann& m);
```

mann.cpp

Die Codedatei zu Klasse Mann

```
#include "Mann.h"
#include <fstream>
using namespace std;

extern ofstream datei;

Mann::Mann(string n, int a): Person(n){
      age = a;
      datei << "+M." << a << endl;
}
Mann::Mann(const Mann& m2): Person("Kai"){
      age = m2.age;
      datei << "+MCopy." << age << endl;
}
Mann::~Mann() {    datei << "-M."  << age << endl;
}
void Mann::Print() const {
      Person::Print();
      datei << ":." << age;
}
void Mann::Setze(string a) {
      datei << "Geht nicht\n";
}
ostream& operator<<(ostream& s, const Mann& m) {
      m.Print();
      return s;
}
```

4. Suchen Sie die Punkte, die die Strecke zwischen $P_1 = (7; 3; 2)$ und $P_2 = (-1; 1; 3)$ dritteln.

Nr. 2	Aufgaben	Punkte
a)	Welche Ausgabe erzeugt der Aufruf von der Funktion `funk1`?	3

```
void funk1(){
        datei << "funk1" << endl;
        Person p("Paul");
        Mann m("Paula", 4);
}
```

b)	Welche Ausgabe erzeugt der Aufruf von der Funktion `funk2`?	3

```
void funk2(){
        datei << "funk2" << endl;
        Person* p1 = new Mann("Fritz", 4);
        Mann* p2 = NULL;
        datei << "Ende\n" << endl;
        p1 = p2;
}
```

c)	Welche Ausgabe erzeugt der Aufruf von der Funktion `funk3`?	4

```
void funk3(){
        datei << "funk3" << endl;
        Person p("Hans");
        Mann m("Paul", 4);
        p = m;
        datei << m << ", " << p << endl;
        p.Setze("Fritz");
        datei << m << ", " << p << endl;
}
```

d)	Wie lautet der dynamische Typ (Laufzeit) der Variablen p1 in Beispiel b) ganz am Ende der Funktion?	1

3. Programmierung

Nr. 3	Aufgabe	Punkte
Hinweis	Es soll das Trinkverhalten eines einzelnen Partygängers protokolliert werden. Erarbeitet wird ein ablauffähiges Programm welches die Klassen **drink** und **partypeople** definiert. Leiten Sie von **drink** die Klassen **bier**, **cocktail** und **softdrink** ab. Leiten Sie von **bier** die Klassen **pils** und **doppelbock** ab. Leiten Sie von **softdrink** die Klassen **cola** und **asascho** ab. Leiten Sie von **cocktail** die Klassen **mojito** und **sexonthebeach** ab. Die Klasse **drink** soll die privaten Attribute **Name** (string), **Alkoholgehalt** (double) und **Volumen** (double) haben. Ergänzen Sie den vorgegebenen Code auf dem Klausurverzeichnis. Sorgen Sie dafür, dass die abgeleiteten Klassen schon in den Konstruktoren durch Verkettung sinnvolle Werte erhalten.	

3. An eine 4m lange Kette, deren Enden wieder 3m auseinander befestigt sind, hängt das Gewicht F = 10 N an einer Stelle, an der die Kette im Winkel von 90° geknickt wird. Berechnen Sie die Seilkräfte.

4. Suchen Sie die Punkte, die die Strecke zwischen $P_1 = (7; 3; 2)$ und $P_2 = (-1; 1; 3)$ dritteln.

a) Erweitern Sie das Menü in `main` und den restlichen Code so, daß Sie ein(e) in `main` definierte Instanz von `partypeople` mit unterschiedlichen `drink`(s) „befüllen" können. Die in den Konstruktoren zu setzenden Default Parameter entnehmen Sie untenstehender Tabelle. 10

Getränk	Milliliter	Alkoholgehalt
Pils	300	0,045
DoppelBock	500	0,075
Cola	300	0
Asascho	400	0
Mojito	200	0,13
SexOnTheBeach	140	0,15

b) Bei jedem Trinkvorgang sollen die privaten Attribute `gesamtalkohol` (double) und `promille` (double) in der Klasse `partypeople` neu berechnet werden. 6

```
Der Gesamtalkohol in Gramm ergibt sich aus Getränkevolumen
in Milliliter * Alkoholgehalt.
Bei der Promilleberechnung nehmen Sie bitte 6000ml Blut
an.
// Medizinisch inkorrekte aber pragmatische Formel:
//
// (Drinkvolumen in ml * Drinkalkoholgehalt) * 100/
//                          6000 ml Blut = Promille
// Also für ein Bier mit 500ml und 4% Alkoholgehalt
// 500 * 0,04 * 100 / 6000 = 0,3 Promille
```

c) Erweitern Sie das Menü in `main` und den restlichen Code so, daß Sie sowohl ein(e) eine Liste der getrunkenen Drinks, als auch den momentanen Promillepegel des `partypeople` ausgeben können. 6

d) Erweitern Sie `MiniMenue` in `main`.cpp mit `try` und `catch` so, daß Sie eine in `partypeople` geworfene (throw) Exception bei der Überschreitung der 2,6 Promille-Grenze gefangen wird. Das Menü soll sich dann beenden und es soll der Hinweis ausgegeben werden, das `partypeople` möge bitte nach Hause gehen. Es folgt die Ausgabe der Getränkeliste und des momentanen Promillepegels bevor sich `MiniMenue` und das Hauptprogramm dann beenden. 6