Übungsaufgaben **Theoretische Physik I**

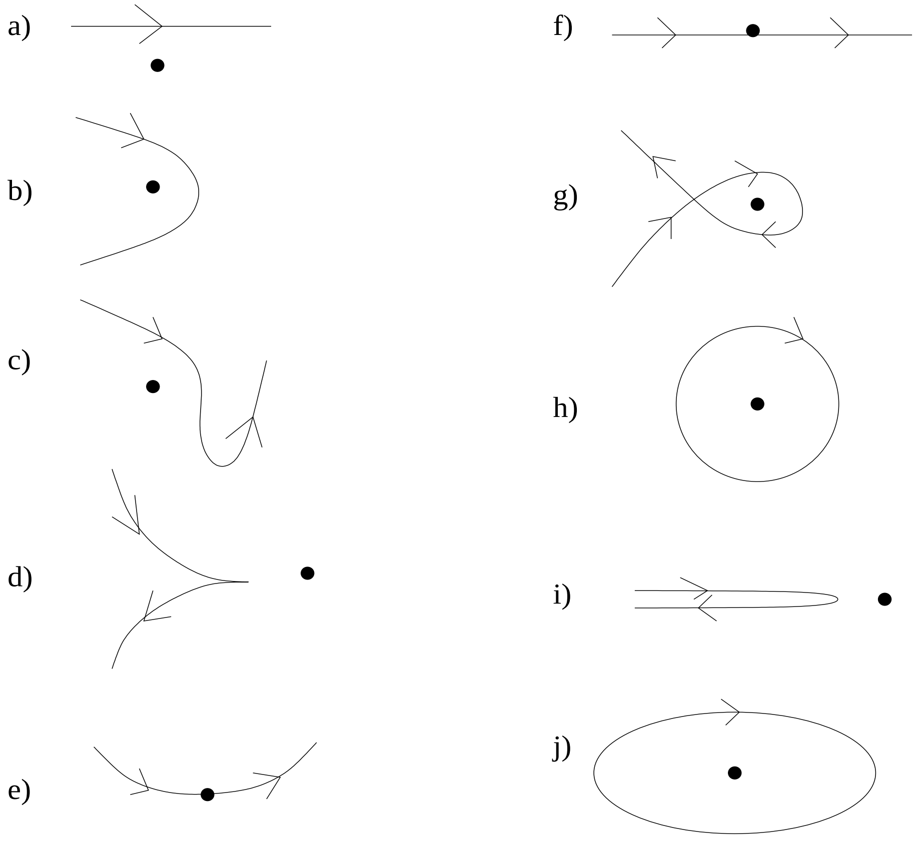
WS 04/05 Prof. Dr. C. Gros

3. Übung Abgabe: Mi.,10.11.04

(bis 12:00, im Briefkasten)

Aufgabe 8

Welche der folgenden Bahnkurven sind mit der Drehimpulserhaltung vereinbar ? $\bullet$ bezeichne den Koordinatenursprung.



Aufgabe 9

Ein Massenpunkt mit der Masse m befindet sich in dem Zentralkraftfeld

$$\vec{F}(\vec{r}) = -\frac{\alpha}{|\vec{r}|^n}\vec{e}_r \qquad (\alpha > 0, n \in \mathbb{R}).$$

Der Massenpunkt werde zur Zeit $t = 0$ am Ort $\vec{r}_0$ mit der Anfangsgeschwindigkeit $\vec{v}_0$ radial vom Kraftzentrum weggeschossen.

a.) Bestimmen Sie die Geschwindigkeit $v = |\vec{v}|$ als Funktion von $r = |\vec{r}|$, r_o, v_0 und n. Bestimmen Sie die minimale Geschwindigkeit $\tilde{v}_0$ (=Fluchtgeschwindigkeit), die der Massenpunkt braucht, um ins Unendliche zu entkommen.

b.) Berechnen Sie die Fluchtgeschwindigkeit $\tilde{v}_0$, falls der Massenpunkt an der Erdoberfläche startet. Auf welchen Radius R_{Ec} müßte die Erdkugel bei konstant gehaltener Masse M_E zusammengedrückt werden, um $\tilde{v}_0 = c$ (=Lichtgeschwindigkeit) zu erhalten?
(Erdradius: $R_E = 6,37 \cdot 10^6$m, Erdmasse: $M_E = 5,98 \cdot 10^{24}$kg, Gravitationskonstante: $G = 6,67 \cdot 10^{-11}\mathrm{m}^3/(\mathrm{kg}\cdot\mathrm{s}^2)$, Lichtgeschwindigkeit: $c = 3 \cdot 10^8$m/s).

Aufgabe 10

Gegeben ist eine Ellipse mit den Halbachsen a und b ($b < a$) und
Brennpunktabstand $2c$. Zeigen Sie :

a.) Es gilt $c^2 = a^2 - b^2$.

b.) Für den Parameter d gilt $d = \frac{b^2}{a} = a\left(1 - \epsilon^2\right)$.

c.) Die Exzentrizität ergibt sich zu $\epsilon = \frac{c}{a}$.

d.) Die Länge der kleinen Halbachse beträgt $b = a\sqrt{1 - \epsilon^2} = \sqrt{a\,d}$.

Aufgabe 11

Tippen Sie folgendes Programm ein und führen Sie es aus.
Definieren Sie die Differenz $A - B$ zwischen Vektoren und λA:
double λ;
Coordinate A, B;
Falls es Unklarheiten gibt, fragen Sie Ihren Assisten oder konsultieren Sie (z.B. im Internet) eine C++ Einführung.

```
/*      November 2004
        Implementation of the Coordinate object */

//includes
#include <iostream.h>   // (input/ouput)
#include <iomanip.h>    // (manipulating input/ouput with `setw()' etc)
#include <math.h>       // (math's commands)
#include <fstream.h>    // (in/output on files)
#include <assert.h>     // (`assert()' calls abort if arg not true)

// global constant
const double pi = 4.0*atan(1.0);   // definition to all digits
//
/* **************************************** */
//
//
// declaration of the class Coordinate

class Coordinate
 {
   double x,y;         // Coordinates x and  y of a particle
   public:             // Member functions (declared inside a class)

   Coordinate(double x_coord,double y_coord)
                       // constructor (initializes the object)
                       // a constructor has the same name and type than the class
   {
     x = x_coord;
     y = y_coord;
   }

   // usual put and get member functions:
   inline double  get_x(){return  x;}
   inline void    put_x(double x_coord) { x=x_coord;}
   inline double  get_y(){return  y;}
   inline void    put_y(double y_coord) { y=y_coord;}

   void  polar_to_cartesian( double radius,double theta)
   {                           //initializes the object from polar coordinates
     x=radius*cos(theta);
     y=radius*sin(theta);
   }

   Coordinate&  operator += (Coordinate a);
};
```

```
inline Coordinate& Coordinate::operator+= (Coordinate a)
{                             // define the += symbol for two vectors
  x+=a.x;
  y+=a.y;
  return *this;
}

Coordinate operator + (Coordinate a, Coordinate b)
{                              // define the + symbol for two vectors
   Coordinate c=a;
   return c+=b;
}

/* ***************************************** */

main()
{
  //Check size of integer, double ,Coordinate.
  cout << "the size of int is        :"<<sizeof(int)<< endl;
  cout << "the size of double is     :"<<sizeof(double)<< endl;
  cout << "the size of Coordinate is :"<<sizeof(Coordinate)<< endl;
  //declaring A as a coordinate and initialize
  //calling the constructor
  //  Coordinate A= Coordinate(2.0,1.0);
  Coordinate A(2.0,1.0);
  //Predefined meaning of operator '='
  Coordinate B=A ;
  Coordinate C=A+B;
  // test output
  cout << " vector A = ("<<A.get_x()<<","<<A.get_y()<<") "<< endl;
  cout << " vector B = ("<<B.get_x()<<","<<B.get_y()<<") "<< endl;
  cout << " vector C = ("<<C.get_x()<<","<<C.get_y()<<") "<< endl;
}
```