OPR Praktikum
Übung: Flugkurve 01

Thomas Mahr

23. Oktober 2019

1 Lernziele

Mit dieser Übung sollen Sie lernen

- zusammengehörende Daten in einer Struktur zu kapseln,
- diese Struktur als Datenelemente in anderen Strukturen zu verwenden,
- Referenzen verwenden.

2 Voraussetzungen

Bis zum Ende des Kapitels *Referenzen* in Teil 1 (*Einführung*).

3 Aufgabenstellung

Das folgende Programm simuliert die Bewegung eines Körpers unter Einfluss der Schwerkraft und Reibungskraft[1]:

```
int main() {
  // Körper:
  float masse = 10; // [kg]
  float xPosition = 0; // [m]
  float yPosition = 0; // [m]
  float xGeschwindigkeit = 10; // [m/s]
  float yGeschwindigkeit = 10; // [m/s]

  // Parametrierung der auf den Körper wirkende Kräfte:
  const float X_BESCHLEUNIGUNG_GRAVITATION = 0; // [m/s^2]
  const float Y_BESCHLEUNIGUNG_GRAVITATION = -9.81; // [m/s^2]
  const float REIBUNGSKOEFFIZIENT = -5; // [kg/s]

  // Sorgt dafür, dass 2 Nachkommastellen angezeigt werden:
  cout << fixed;
  cout.precision(2);

  // Simulation:
  const float dt = 0.1; // [s]
```

[1] Stoke'sche Reibung: Die Reibungskraft ist für kleine Geschwindigkeiten proportional zur Geschwindigkeit.

```
    for(;;) {
      cout << "(" << xPosition << "," << yPosition << ")" << endl;

      // Berechnung der Beschleunigung aus der Kraft
      float xBeschleunigung = xGeschwindigkeit * REIBUNGSKOEFFIZIENT
          / masse + X_BESCHLEUNIGUNG_GRAVITATION;
      float yBeschleunigung = yGeschwindigkeit * REIBUNGSKOEFFIZIENT
          / masse + Y_BESCHLEUNIGUNG_GRAVITATION;

      // Ermittlung der neuen Geschwindigkeit
      xGeschwindigkeit += dt * xBeschleunigung;
      yGeschwindigkeit += dt * yBeschleunigung;

      // Ermittlung der neuen Position
      xPosition += dt * xGeschwindigkeit;
      yPosition += dt * yGeschwindigkeit;

      if(yPosition<=0) {
        break;
      }
    }
  }
```

Schreiben Sie das Programm um:

- Fassen Sie fachlich zusammenhängende Daten in den Strukturen `Vektor` und `Koerper` zusammen.
- Verwenden Sie Funktionen zur Addition zweier Vektoren, Multiplikation eines Vektors mit einem Skalar, Ausgabe eines Vektors auf der Konsole, Bewegung eines Körpers und Ausgabe eines Körpers auf der Konsole.
- Übergeben Sie Funktionsargumente per (konstanter) Referenz.
- Verwenden Sie in Ihrer Lösung die folgende `main()`-Funktion:

```
int main() {
  // Körper:
  Koerper koerper;
  koerper.masse = 10;
  koerper.position = { 0, 0 };
  koerper.geschwindigkeit = { 10, 10 };

  // Parametrierung der auf den Körper wirkende Kräfte:
  const Vektor BESCHLEUNIGUNG_GRAVITATION = {0, -9.81}; // [m/s^2]
  const float REIBUNGSKOEFFIZIENT = -5; // [kg/s]

  // Sorgt dafür, dass 2 Nachkommastellen angezeigt werden:
  cout << fixed;
  cout.precision(2);

  // Simulation:
  const float dt = 0.1;
  for(;;) {
    cout << "Position des Körpers: ";
    ausgabeKoerper(koerper);
```

```
    cout << endl;

    // Bestimmung der Kraft
    Vektor reibungskraft = mulVektor(koerper.geschwindigkeit,
       REIBUNGSKOEFFIZIENT);
    Vektor anziehungskraft = mulVektor(BESCHLEUNIGUNG_GRAVITATION,
       koerper.masse);
    Vektor gesamtkraft = addVektor(reibungskraft,anziehungskraft);

    // Ermittlung der neuen Position und neuen Geschwindigkeit
    bewegeKoerper(koerper,gesamtkraft,dt);

    if(koerper.position.y<=0) {
      break;
    }
  }
}
```