Norges teknisk–naturvitenskapelige universitet
Institutt for datateknikk og informasjonsvitenskap

TDT4102 Prosedyre og Objektorientert programmering Vår 2014

Løsningsforslag øving 2

Frist: DD.MM.YYYY

Mål for denne øvingen:

- Lære om funksjoner
- Lære forskjellene mellom vanlige variabler og pekere
- Lære å skrive koden din i flere filer
- Lære å bruke nyttig matematiske funksjoner som sin og cos

Generelle krav:

- Bruk de eksakte navn og spesifikasjoner som er gitt i oppgaven
- Det er valgfritt om du vil bruke en IDE (Visual Studio, Xcode), men koden må være enkel å lese, kompilere og kjøre
- Dersom noe er uklart eller trenger en bedre forklaring ta kontakt med en studass på sal.

Anbefalt lesestoff:

- Kapittel 3 & 4, Absolute C++ (Walter Savitch)

1 Funksjonshoder (10%)

a) Lag et funksjons hode som returnerer akselerasjonen i y-retning (oppover). Akselerasjonen i y-retning er normalt et desimaltall. Denne funksjonen skal hete "acclY".

Løsning:

```
double acclY();
```

b) Skriv et funksjonshode som regner ut farten i y-retning (oppover). Denne funksjonen tar inn to flyttall (double): startfart og tid. Til slutt returnerer funksjonen farten som et flyttall. Funksjonen skal hete `"velY"`.

Løsning:

```
double velY(double startVel, double t);
```

c) Skriv et funksjonshode som regner ut farten i y-retning (oppover) med integrasjon. Denne funksjonen tar inn to flyttall (double): startfart og tid. Til slutt returnerer funksjonen farten som et flyttall. Funksjonen skal hete "velIntY".

Løsning:

```
double velIntY(double startVel, double t);
```

d) I denne deloppgaven skal vi lage et sett av funksjonshoder for utregning av posisjon både i X- og Y-retning.

Løsning:

```
double posX(double startVel, double t);
double posY(double startVel, double t);

double posIntX(double startVel, double t);
double posIntY(double startVel, double t);
```

e) Skriv et funksjonshode som tar inn tid i sekunder og ikke returnerer noe. Denne skal hete `"printTime"`.

Løsning:

```
void printTime(double t);
```

f) Skriv et funksjonshode som tar in startfarten i y-retning og returnere flytiden i sekunder. Denne skal hete `"flightTime"`.

Løsning:

```
double flightTime(double startVelY);
```

2 Implementer funksjoner (20%)

a) I denne oppgaven skal du implementere funksjonen fra oppgave 1a. Denne funksjonen returnerer akselerasjonen i y-retning (oppover). Til vanlig er akelerasjonen i y-retning -9.81m/s (gjenstander trekkes mot bakken).

Løsning:

```
double acclY(){
    return -9.81;
}
```

b) Denne oppgaven skal implementere funksjonen fra oppgave 1b (fart i y-retning).

Løsning:

```
double velY(double startVel, double tid){
    return startVel + acclY() * tid;
}
```

c) Nå skal vi implementere funksjonen fra oppgave 1c (fart i y-retning med integral).

Løsning:

```
double velIntY(double startVel, double tid){
    double accVelocity = startVel;
    double curTime = 0;
    while (curTime < tid){
        accVelocity += acclY()*intStep;
        curTime += intStep;
    }
    return accVelocity;
}
```

d) I denne oppgaven skal vi lage / implementere funksjonene som regner ut posisjonen i X- og Y-retning (oppgave 1d).

Løsning:

```
double posX(double startVel, double tid){
    double position = startVel*tid;
    return position;
}
double posY(double startVel, double tid){
    double position = startVel*tid + 0.5 *pow(tid,2.0) * acclY();
    return position;
}

double posIntX(double startVel, double tid){
    double accPosition = 0;
    double curTime = 0;
    while (curTime < tid){
        accPosition += startVel*intStep;
        curTime += intStep;
    }
    return accPosition;
}
double posIntY(double startVel, double tid){
    double accPosition = 0;
    double curTime = 0;
    while (curTime < tid){
        accPosition += velIntY(startVel,curTime)*intStep;
        curTime += intStep;
    }
    return accPosition;


}
```

e) Implementer funksjonen `"printTime"` (oppgave 1e). Vi ønsker å dele opp sekundene i timer, minutter og sekunder (sekunder kan evt være et desimaltall) for så å skrive dette til skjerm.

Løsning:

```
  void printTime(double tid){
      int hours = (int)tid/3600;
      int minutes = ((int)tid - 3600*hours)/60;
      double seconds = tid - hours*3600 - minutes * 60;
      cout << hours << " hours, " << minutes << " minutes and "
              << seconds << " seconds" << endl;
  }
```

f) Implementer funksjonen "flightTime" (oppgave 1f).

Løsning:

```
double flightTime(double startVelY){
    return -startVelY/acclY()*2.0;
}
```

3 Verifiser at funksjonene fungerer (10%)

a) Forsikre deg om at programmet kompilerer.

b) Test hver metode fra main funksjon.

Løsning (for et fullstendig oppsett se vedlagt kode):

```
result = posX(10.0,3.0);
solution = 30;
deviation = pow(result-solution,2.0);
if (deviation > maxError){
    cout << "PosX() is malfunctioning (returned: "
        << result << " expected: " << solution << ")" <<  endl;
}

result = posIntX(10.0,3.0);
solution = 30;
deviation = pow(result-solution,2.0);
if (deviation > maxError){
    cout << "PosIntX() is malfunctioning (returned: "
        << result << " expected: " << solution << ")" <<  endl;
}


result = flightTime(30.0);
solution = 6.116;
deviation = pow(result-solution,2.0);
if (deviation > maxError){
    cout << "FlightTime() is malfunctioning (returned: "
        << result << " expected: " << solution << ")" <<  endl;
}
```

4 Implementer funksjoner (20%)

a) I denne oppgaven skal vi implementere følgende funksjoner:

```
void getUserInput(double *theta, double *absVelocity);
double getVelocityX(double theta, double absVelocity);
double getVelocityY(double theta, double absVelocity);
void getVelocityVector(double theta, double absVelocity,
        double *velocityX, double *velocityY);
```

Løsning:

```
void getUserInput(double* angle, double* absVelocity){
    cout << "Please enter a angle for \
            the cannon angle:" << endl;

    cin >> (*angle);
    while(*angle < 0.1){
    cout << "The angle you chose was too small, \
            please select a larger one." << endl;
    cin >> (*angle);
    }

    cout << "Please enter a velocity for \
             the cannon projectile:" << endl;
    cin >> (*absVelocity);
    while(*absVelocity < 0.1){
    cout << "The velocity you chose is too \
             small, try again." << endl;
    cin >> (*absVelocity);
    }

}
double getVelocityX(double angle, double absVelocity){
    return cos(angle)*absVelocity;
}
double getVelocityY(double angle, double absVelocity){
    return sin(angle)*absVelocity;
}

void getVelocityVector(double angle, double absVelocity,
        double *velocityX, double *velocityY){
    *velocityX = getVelocityX(angle, absVelocity);
    *velocityY = getVelocityY(angle, absVelocity);
}
```

b) Implementer funksjonen "getDistanceTraveled"

Løsning:

```
double getDistanceTraveled(double velocityX, double velocityY){
    double fTime = flightTime(velocityY);
    double distanceTraveled = posX(velocityX,fTime);
    return distanceTraveled;
}
```

c) Implementer funksjonen "optimalAngleForMaxDistance"

Løsning:

```
double optimalAngleForMaxDistance(double absVelocity){
    double PI = 3.14;
    double bestAngle = 0.001;
    double bestDistance = 0;
    double velocityX, velocityY;
    for (double angle = 0.001; angle < PI/2; angle+= 0.001){

        getVelocityVector(angle, absVelocity, &velocityX, &velocityY);
        double distance = getDistanceTraveled(velocityX,velocityY);
        if(distance > bestDistance){
            bestAngle = angle;
            bestDistance = distance;
        }
    }
    return bestAngle;
}
```

d) Implementer funksjonen "targetPractice".

Løsning:

```
    double targetPractice(double distanceToTarget,
            double velocityX, double velocityY){
        double error = distanceToTarget -
                getDistanceTraveled (velocityX, velocityY);
        return pow(error,2.0);
    }
```

5 Verifiser at funksjonene fungerer (10%)

a) Verifiser at programmet kompilerer

b) Lag en kodesnutt i "main()" som tester funksjonene

Løsning: Funksjonene kan testes på sammme måte som er gjort i løsningsforslaget for oppgave 3b.

6 Større program (20%)

a) Legg til funksjonen "playTargetPractice" til biblioteket og legg til kode i "main()" for å kjøre denne funksjonen.

Løsning:

```
void playTargetPractice(){
    cout << "*******Playing target practice!******" << endl;
    cout << "The goal of this game is to hit the target, you will be presented with
    a target at some random distance from you. In order to hit a target you
    must specify an angle and a initial velocity for
    the cannonball." << endl << endl;
    double distanceToTarget = 20.0;
    double maxError = 2.0;
    bool win = false;

    cout << "************************************" << endl;
    cout << "****Target is placed at " << distanceToTarget <<  "m range****" << endl;
    cout << "************************************" << endl;
    cout << "************Happy hunting!***********" << endl;
    cout.precision(2);
    cout.setf(ios::fixed);
    for (int chance = 0; chance < 10; chance++){

        cout << "************************************" << endl;
        cout << "***************Round: " << chance+1  << "**************" << endl;
        cout << "************************************" << endl;
        double angle, absVelocity, velocityX, velocityY;
        getUserInput(&angle, &absVelocity);
        getVelocityVector(angle, absVelocity, &velocityX, &velocityY);

        double error = targetPractice(distanceToTarget, velocityX, velocityY);
        double distanceTraveled = getDistanceTraveled(velocityX,velocityY);
        if (error < maxError){
            win = true;
            break;
        }
        cout << "Your shot went: " << distanceTraveled << "m" << endl;
        cout << "Distance from target: " << sqrt(error) << "m" <<  endl;
        if (distanceTraveled > distanceToTarget){
            cout << "You overshot the target! A little less force next time maybe?" << endl;
        }
        else{
            cout << "You didnt reach the target. Some more force next time?" << endl;
        }
    }
    if (win){
        cout << "Congratulations you won!" << endl;
    }
    else{
        cout << "Sorry, you lost." << endl;
    }
}
```