Kurserne 02312-14 Indledende Programmering, 02313 Udviklingsmetoder til IT-Systemer og 02315 Versionsstyring og Testmetoder.

Lektion 2

Opgave 2

Gruppe nr. 15.

Navn og studienummer:

Opgave 1:
Karl Emil Jeppesen s180557
Alfred Röttger Rydahl s160107
Said Abdullahi s185013

Opgave 2:
Søren Poulsen s180905
Rasmus Sander Larsen s185097
Noah F. M. Hamza s185084

Timeregnskab

Navn	Timer
Karl Emil	1
Alfred	1
Saaid	1
Søren	1
Rasmus	1
Noah	1

Kravspecifikation

Denne opgave går ud på at designe, implementere og teste et program der kan beregne faldtiden og sluthastigheden for en faldende sten ud fra inputtet højde. Der tages ikke højde for luftmodstand.

Design

Det generelle design er at brugeren promptes til at indtaste et input.

Det testes om dette input er brugbart – det er et tal og større eller lig med nul.

Inputtet anvendes til at beregne faldtid og sluthastighed og disse værdier udskrives til brugeren.

Det søges at designe programmet således at det genstarter ved 'forkert' input.

Implementering

Der skal bruges en klasse som kaldes 'FallingObject'. Der skal desuden bruges en 'main' funktion til at køre programmet.

Det ønskes, at programmet 'genstartes' ved forkert input. Dette gøres i praksis ved at programmets handlinger findes i en 'constructor' som kaldes i 'main' funktionen og desuden kalder på sig selv ved forkert input. Constructoren kaldes ligeledes 'FallingObject()'.

Hvis inputtet er et tal og godkendes af 'try/catch' funktionen (NumberFormatException) testes det om tallet er større eller lig med nul gennem en if/else statement.

Slutteligt beregnes faldtid og sluthastighed hvis tallet er større end nul.

Faldtiden er givet ved $\sqrt{\frac{2 \cdot højde}{g}}$

Sluthastigheden er givet ved $tid \cdot g$

Højde er udtryk for højden hvorfra objektet falder og g er udtryk for tyngdeacceleration (9,801m/s^2)

Begge tal afrundes til to decimaler gennem Math.round();

Test

Positive test
Første iteration af programmet var succesfuldt i positive test. Det gjorde hvad det skulle når input var af den ønskede karakter. Dog var output ofte med mange decimaler.

Da input og output er doubles blev output udskrevet med mange decimaler hvilket ikke anses ønskværdigt. Dette problem blev afhjulpet ved at anvendes Math.round();

Negative test
Første iteration af programmet skelnede ikke mellem positive og negative tal. Dette gav muligheden for negative input og dermed output. Denne fejl blev rettet med en if/else statement med condition (input >= 0).

Kurserne 02312-14 Indledende Programmering, 02313 Udviklingsmetoder til IT-Systemer og 02315 Versionsstyring og Testmetoder.

Programmet gav en rød fejlmeddelelse ved input af bogstaver. Denne fejl forsøgtes rettet gennem if/else og ASCII men da det viste sig en kompliceret løsning blev try/catch implementeret. Denne fanger alle input fejl som følge af indtastninger der ikke er tal.

Meget store og små tal håndteres udmærket af programmet da det scanner hele inputtet som en string og derefter konverterer til double. Det anses, at opløsningen i doubles er tilstrækkelig til formålet. Om nødvendigt kan mængden af decimaler ændres.

Ved 400 km (400.000 m) er gravitationen ca. 90% af g, hvorfor tal herover ikke kan beregnes nøjagtigt.

Det er med seneste version af programmet ikke lykkes at få det til at ’crashe’.