Startdocument for Parental Contribution

Startdocument of **Victor Peters**. Student number **4985664**.

Problem Description

An elementary school calculates the parental contribution using the following method: €38,- is charged for children under the age of 6. €50,- is charged for children aged 7 to 9. €65,- is charged for children above the age of 9.

Input & Output

In this section the in- and output of the application will be described.

Input

In the table below all the input (that the user has to input in order to make the application work) are described.

Case	Type	Conditions
Date of Birth	`DateTime`	not empty
Name Of Child	`String`	not empty
Age Of Worker	`int`	0 < `number`
Kilometers	`int`	0 < `number`

Output

Case	Type
Parental contribution per child	`double`
Total contribution	`double`
Age of child	`int`
Number of children per age category	`double`

Calculations

Case	Calculation
Age of child	The current date subtracted from the date of birth subtracted by 1 depending on the current month
Total parental contribution	adding the price of each child.

Remarks

- Input will be validated.

Class Diagram

Class Diagram

GUI design

GUI design

Testplan

In this section the testcases will be described to test the application.

Test Data

In the following table you'll find all the data that is needed for testing.

School

ID	Input	Code
`deKubus`	name: De Kubus	`new School("De Kubus")`

Category

ID	Input	Code
`<6`	name: youngerThan6 price: 38	`new Category("youngerThan6", 38)`
`6-9`	name: between6and9 price: 50	`new Category("between6and9", 50)`
`>9`	name: olderThan9 price: 65	`new Category("olderThan9", 65)`

Student

ID	Input	Code
`nick`	name: Nick yearOfBirth: 2012 monthOfBirth: 1 dayOfBirth: 1	`new Student("Nick", 2012, 1, 1)`

ID	Input	Code
`jantje`	name: Jantje yearOfBirth: 2018 monthOfBirth: 12 dayOfBirth: 1	`new Student("Jantje", 2018, 12, 1)`
`thomas`	name: Thomas yearOfBirth: 2015 monthOfBirth: 1 dayOfBirth: 1	`new Student("Thomas", 2015, 1, 1)`
`jordi`	name: Jordi yearOfBirth: 2016 monthOfBirth: 1 dayOfBirth: 1	`new Student("Jordi", 2016, 1, 1)`

Add student to school

School	Code
`deKubus`	`addStudent(new Student("Nick", 2012, 1, 1))`
`deKubus`	`addStudent(new Student("Jantje", 2018, 12, 1))`

Add student to category

Category	Code
`<6`	`addStudent(new Student("Jantje", 2018, 12, 1))`
`>9`	`addStudent(new Student("Nick", 2012, 1, 1))`

Test Cases

In this section the testcases will be described. Every test case should be executed with the test data as starting point.

#1 Get a student's age

Determinining a student's age.

Step	Input	Action	Expected output
1	`jantje`	`calculateAge()`	3
2	`nick`	`calculateAge()`	10

#2 Get contribution per child

Get the contribution per child.

Step	Input	Action	Expected output
1	`nick`	`getContribution()`	65
2	`thomas`	`getContribution()`	50

#3 Calculate total contribution

Calculate the total contribution.

Step	Input	Action	Expected output
1	`deKubus`	`calculateTotalContribution()`	103

#4 Get youngest student

Get the younngest student.

Step	Input	Action	Expected output
1	`deKubus`	`getYoungestStudent()`	`Jantje`

Using the program

This section will describe what happens when you add new students to a school.

1 Calculate total contribution after adding new students

Step	Input	Action	Expected output
1	`deKubus`	`Add Nick, aged 10, to the list of students`	
2	`deKubus`	`Calculate the total contribution`	65
3	`deKubus`	`Add Jantje, aged 3, to the list of students`	
4	`deKubus`	`Calculate the total contribution`	103

2 Get youngest student

Step	Input	Action	Expected output
1	`deKubus`	`Add Nick, aged 10, to the list of students`	
2	`deKubus`	`Get the youngest student`	`Nick`
3	`deKubus`	`Add Jantje, aged 3, to the list of students`	
4	`deKubus`	`Get the youngest student`	`Jantje`