Exam PCS2 2018, Jan 19

Time: 9:30 - 12:00 h (150 minutes).

Admitted resources:

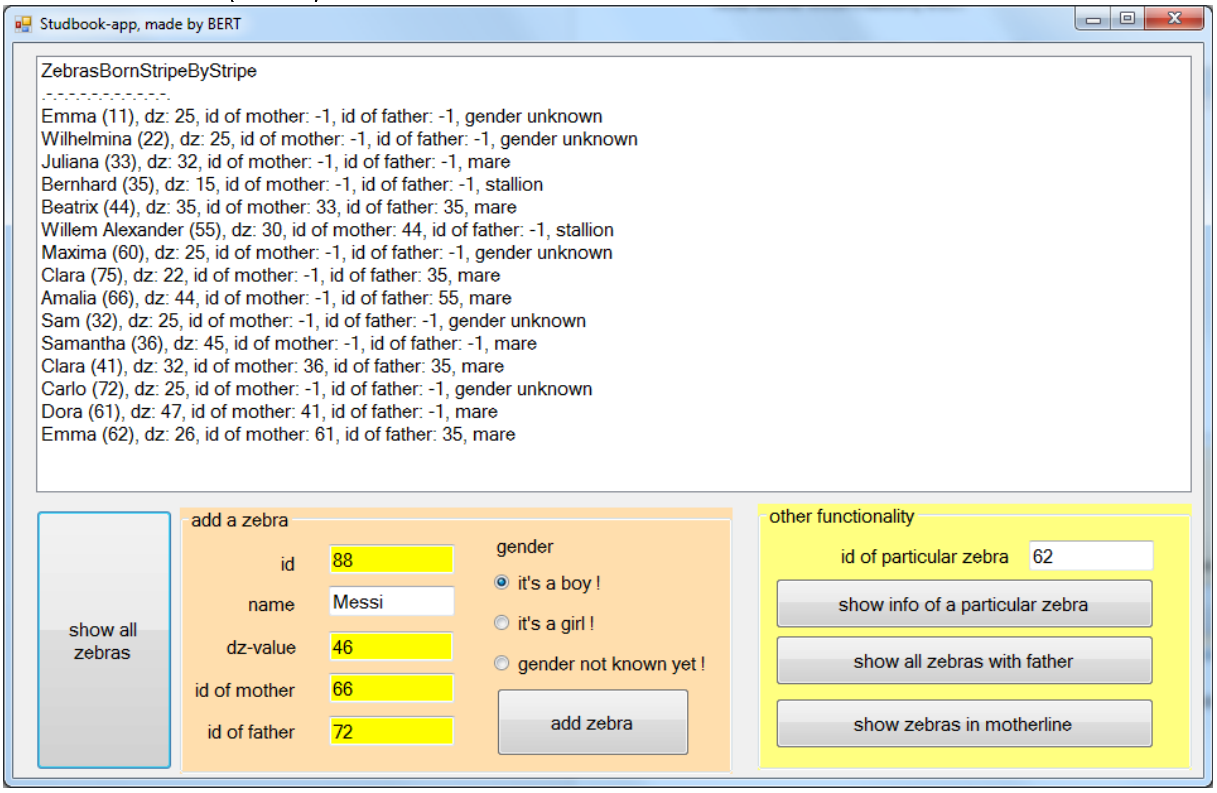
- You are allowed to use everything on paper (books, notes, etc.) and on your laptop, but only what you bring in: you are not allowed to borrow something from someone else.
- During the exam it is not allowed to use the network. You should make the exam yourself: so no communication with MSDN or google for help and no communication with other students, like using Facebook, e-mail, Skype, Dropbox, mobile phone or whatever.

+-

Introduction.

A studbook is an official list of animals within a specific breed. For most animals their parents are known.
we need an application to handle the studbook information of zebras . For every zebra the studbook should store some information. The application should have some functionality, like adding a zebra to the system, find some information, etc..

The user interface (Form1) looks like:



```
ZebrasBornStripeByStripe
-.-.-.-.-.-.-.-.-.-.-.-.
Emma (11), dz: 25, id of mother: -1, id of father: -1, gender unknown
Wilhelmina (22), dz: 25, id of mother: -1, id of father: -1, gender unknown
Juliana (33), dz: 32, id of mother: -1, id of father: -1, mare
Bernhard (35), dz: 15, id of mother: -1, id of father: -1, stallion
Beatrix (44), dz: 35, id of mother: 33, id of father: 35, mare
Willem Alexander (55), dz: 30, id of mother: 44, id of father: -1, stallion
Maxima (60), dz: 25, id of mother: -1, id of father: -1, gender unknown
Clara (75), dz: 22, id of mother: -1, id of father: 35, mare
Amalia (66), dz: 44, id of mother: -1, id of father: 55, mare
Sam (32), dz: 25, id of mother: -1, id of father: -1, gender unknown
Samantha (36), dz: 45, id of mother: -1, id of father: -1, mare
Clara (41), dz: 32, id of mother: 36, id of father: 35, mare
Carlo (72), dz: 25, id of mother: -1, id of father: -1, gender unknown
Dora (61), dz: 47, id of mother: 41, id of father: -1, mare
Emma (62), dz: 26, id of mother: 61, id of father: 35, mare
```

show all zebras

add a zebra

id	88
name	Messi
dz-value	46
id of mother	66
id of father	72

gender
[✓ it's a boy !]
it's a girl !
gender not known yet !

add zebra

other functionality

id of particular zebra: 62

show info of a particular zebra
show all zebras with father
show zebras in motherline

In the window displayed above you see a listbox, a "show all zebras"-button and two groupboxes.

The listbox, displays on the first line the name of the studbook, then a line like `".-.-.-.-.-.-.-.-.-.-.-."` , then on the next lines information of the zebras.

On the bottom side on the left you see a button to show all zebras. This button works already.
In the bottom side middle (the "add a zebra" groupbox) the user can type information about a zebra and then add a zebra with this data to the studbook (to be implemented by you). In the bottom side right (the "other functionality" groupbox) you see some buttons (also need to be implemented by you).

Now open the startup-project . You can see that there is already a user interface (Form1) with a lot of controls on it (see the picture above).
The project has a class Studbook and an enumeration Gender. It also needs a class Zebra. It will be your task to finish the app.
Assignment 1 is about the Zebra-class; assignment 2 about the Studbook-class and assignment 3 is about Form1.

Assignment 1: The "Zebra" class (8 + 10 + 6 = 24 points)

Add a Zebra-class to the project and make sure it has the following functionality:

Assignment 1a:
For every Zebra we need to register the following data:

- a unique identity-number (a positive integer),
- its name (a string),
- its dz-value (in case you never heard of it: we will not explain it now, but it must be an integer value at least 10 and at most 50. When trying to assign a value bigger than 50, you need to store the value 50; when trying to assign a value lesser than 10, store the value 10)
- the identity-number of its mother (a positive number if known, otherwise -1)
- the identity-number of its father (a positive number if known, otherwise -1)
- its gender ("stallion" means "male"; "mare" is "female" and sometimes a zebra is born and we do not know its gender yet: "unknown"). To store its gender, you MUST use the enum Gender, which is already in the start-up-project.

Add fields and/or properties and/or methods to your Zebra-class to be able to store the above mentioned data.

Assignment 1b:
Implement 3 constructors for this class. The constructors should create the Zebra-object according to the values of their input parameters:

- `int id, String name`
  This constructor creates a Zebra-object with identity-number and name as specified by the parameters. Make sure the identity-number of the mother and father is initialized as -1, the dz as 25 and the gender as unknown;
- `int id, String name, int motherId`
  This constructor does the same as the one before, but now the identity-number of the mother should be as specified by the parameter
- `int id, String name, int dz, int motherId, int fatherId, Gender g`
  This constructor creates a Zebra-object according to the values for the parameters. Be aware that the dz should be at least 10 and at most 50 (in case it is bigger than 50 you need to store the value 50; in case it is less than 10, store the value 10).

Assignment 1c:
Add a method `getInfo()` to the Zebra-class. This method should return a string with information about its data (for instance, as pictured in the screendump on the first page).

Assignment 2: The “Studbook” class (6 + 6 + 8 + 10 + 10 = 40 points)

The class Studbook has 2 data-fields, one to hold its name and one to hold a collection of Zebra-objects. In the start-up-project it is a list, but if you prefer an array or whatever, feel free to change it.

Assignment 2a:
Implement the constructor `public Studbook(String name)`
as specified in the 3 slashes /// above this constructor.

Assignment 2b (adding a zebra, first version):
Implement the method `public bool AddZebra(Zebra z)`
as specified in the 3 slashes /// above this method.

Assignment 2c (adding a zebra, improved version):
This assignment improves the former version. If you manage to do assignment 2b but not 2c, you still get the points for assignment 2b.
Now there are some more restrictions for adding a new zebra.
In case there is no zebra yet in the collection of zebras with the same identity-number, the following extra rules should be applied:

- if the value for the father-id is -1, you should use -1 as father-id of the new zebra.
- if the value for the father-id is positive and matches a male zebra in the collection, you should use that father-id, otherwise use -1 as father-id.
- if the value for the mother-id is -1, you should use -1 as mother-id of the new zebra.
- if the value for the mother-id is positive and matches a female zebra in the collection, you should use that mother-id, otherwise use -1 as mother-id.

Implement these extra rules in the method `public bool AddZebra(Zebra z)`.

Assignment 2d:
Implement the method `public List<Zebra> GetZebrasWithFather(int idFather)`
as specified in the 3 slashes /// above this method.

Assignment 2e:
Implement the method `public List<Zebra> GetAncestorsInMotherline(int id)`
as specified in the 3 slashes /// above this method.

Assignment 3: The “Form1” class (4 + 6 + 8 + 8 + 10 = 36 points)

Assignment 3a:
The class Form1 must have a variable `myStudbook` of type Studbook to store information of the zebras. Add such a variable.
At startup, that variable should be created. Choose a name for your studbook.
At the top of the window in the example, you see in the title-bar the text `"Studbook-app, made by BERT"`. Have a text in the title bar like this text, but with your name in it.
By the way, you may use the method

```
  private void AddTestingStuff()
```

if you like, but you might also skip it. It is just in it to help you test your code.

The button with text "show all zebras" is already implemented.

Assignment 3b:
Clicking the button with text "show info of a particular zebra" should show information of a particular zebra, based on an identity-number as inputted in the textbox after "id of particular zebra". Clicking this button results in showing information about the zebra with the inputted identity-number on the screen, in case there is such a zebra. If no zebra with that identity-number exists, show an appropriate message (how, that's up to you!).
Implement the method `btnShowInfoOf1Zebra_Click(. . .)`.

Assignment 3c:
The user types information of a zebra in the textboxes (for id, name, etc) in the bottom side middle.
You may assume the user types in the yellow textboxes texts that can be converted to an integer.
You may also assume that the required identity-numbers are positive or equal to -1.
With selecting one of the radiobuttons the user indicates the gender of the zebra.

Clicking the button with text " add zebra" should result in:

1. if there is already a zebra in your collection of zebras with the same identity-number, nothing is added to the collection of zebras
2. if there is no zebra yet in the collection of zebras with the same identity-number, a new zebra-object is created and added to the collection of zebras.

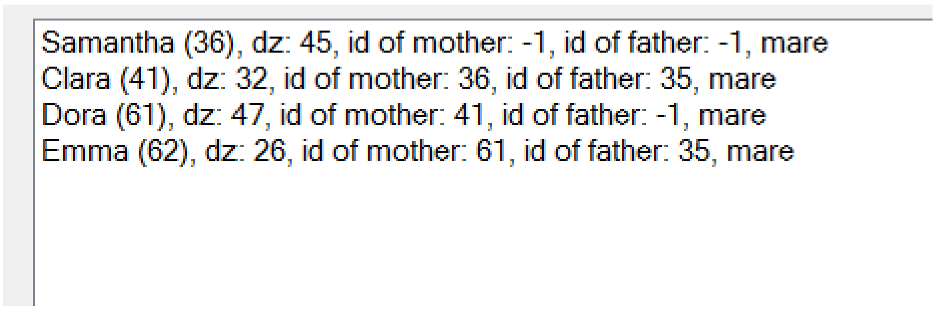
In both cases an appropriate message should be shown telling the user if a new zebra is added to the collection or not.
Implement the method `btnAddZebra_Click(. . .)`.

> An example*: consider the before mentioned screendump: clicking this button results in creating a new Zebra with id 88, name Messi, its dz is 46, motherid is 66, fatherid is 72 and it's gender is stallion.*

Assignment 3d:
The user has to input an identity-number in textbox after "id of particular zebra".
Clicking the button with text "show all zebras with father" should show information of all zebras, of which the fathered is equal to the number in the textbox tbOthers.
Implement the method `btnShowZebrasWithFather_Click(. . .)`.

Assignment 3e:
Clicking the button with text "show zebras in motherline" should show information of the motherline of a particular zebra, based on an identity-number, as inputted in textbox after "id of particular zebra".
After clicking this button, in the listbox you see information about this particular zebra, its mother, its mother's mother, its mother's mother's mother, etc., as long as the mother is known.
The bottom line in the listbox shows information about this particular zebra;
the line above the bottom line shows information about this particular zebras mother;
the line above the former mentioned line shows information about this particular zebras mother's mother, etc.

> An example*: consider the before mentioned screendump: The user inputted 62 in the textbox. clicking this button results in the following info in the listbox*:



```
Samantha (36), dz: 45, id of mother: -1, id of father: -1, mare
Clara (41), dz: 32, id of mother: 36, id of father: 35, mare
Dora (61), dz: 47, id of mother: 41, id of father: -1, mare
Emma (62), dz: 26, id of mother: 61, id of father: 35, mare
```

Implement the method `btnShowZebrasInMotherline_Click(. . .)`.

END of pcs2-exam.