הנדסת תוכנה 094129

תרגיל בית 3

תכנות מודולרי, תרגול ירושה ופולימורפיזם.

תאריך אחרון להגשה: 24/06/2019 בשעה 23:55

מתרגל אחראי: רום גוטמן

נושא התרגיל: מערכת מידע לגן החיות האזורי - המשך מתרגיל מספר 2:

תיאור המערכת:

במסגרת שיתוף פעולה עם גן החיות האזורי, התבקשתם לבנות מערכת מידע על משפחות החיות בפארק.

לכל סוג של חיה נרצה להחזיק מידע כללי על האוכלוסיה ומידע פרטי על כל אחד מהפרטים.
המידע אותו נרצה להחזיק על האוכלוסיה הוא: מספר הפרטים באוכלוסיה, וכן את כל הפרטים הקיימים באוכלוסיה.
המידע אותו נרצה להחזיק על כל פרט הוא, השם שלה (לכל חיה יש שם ייחודי), המין של החיה, האם החיה הגיעה לגיל הבגרות ויכולה להעמיד צאצאים. נרצה להיות מסוגלים לגשת מהחיה אל האמא והאבא שלה, וכן אל הצאצאים שלה, בצורה ישירה (רק אם הם בין החיים).

בנוסף (**תוספת מתרגיל בית 2**) נשמור את סוג החיה: אריה, ברווז או אחד ממשפחת בעלי החדק (פיל או ממותה) וכיוון שכל חיה מגיעה לפרקה בגיל שונה, נשמור גם את גיל החיה. כמו כן, לכל סוג חיה יש צליל מיוחד משלה.

מידי פעם לגן החיות מגיעה לגן החיות חיה חדשה, אשר מגיעה משושלת חדשה (ואף סוג חדש).

אנו נדרשים להיות מסוגלים לתמוך בפעולות הבאות:

1. אתחול של גן חיות הכולל שני פרטים, זכר ונקבה.
   1.1. הפרטים שיווצרו באתחול תמיד יהיו במצב שהם לא הגיעו עדיין לגיל בגרות.
   1.2. שני הפרטים הראשונים באוכלסייה יהיו מאותו הסוג.
2. רבייה עם פרט אחר באוכלוסיה מהמין והסוג המתאים.

3. הרדמה של פרט.
   3.1. אם לצאצא שלא הגיע לגיל הבגרות מתו שני ההורים, אז הוא גם ימות.
4. "טיול בעץ המשפחה של פרט"
5. השמעת צליל על ידי החיה.
6. הוספת חיה חדשה.
   6.1. לחיה החדשה שנוסיף אין קשר לשושלת הקיימת בגן החיות שלנו

מבני הנתונים של התוכנה:

כחלק מהפתרון עליכם לממש את המחלקות הבאות, בהתאם לממשק המתואר להלן:

<u>דרישות מחלקה Zoo:</u>

Method Name	Description
Zoo(const string& maleName, const string& femaleName, const string& type);	בונה המחלקה, מקבל את שמות שני הפרטים המייסדים של האוכלוסיה ומחרוזת המייצגת את סוג החיה. const string& maleName: שם הפרט הזכר הראשון const string& femaleName: שם הפרט הנקבה הראשונה const string& type: סוג החיה (ראו מטה)
~Zoo();	הורס המחלקה
bool animalReproduce(const string& firstParentName, const string& secondParentName,const string& name_,Gender gender);	מתודת הרבייה: מקבלת כקלט את שמות שני הפרטים שרוצים להתרבות, שם הצאצא והמין שלו. המתודה מחזירה true אם הרביה הצליחה, וfalse אם הרביה נכשלה. רבייה מוצלחת מתרחשת כאשר: 1. שני הפרטים הם בגיל הבגרות 2. שני הפרטים ממין שונה 3. שני הפרטים מאותו הסוג
bool euthanize(const string& name);	מתודת ההרדמה: מקבלת שם של פרט באוכלוסיה וממיתה אותו. אם שם הפרט אינו נמצא בגן החיות, התהליך נכשל והמתודה תחזיר false. שימו לב שלפי חוקי הטבע אם לחיה שלא הגיעה לגיל בגרות מתו שני ההורים אז גם היא מתה.
void growUp() const;	**(בשונה מתרגיל בית 2)** המתודה מגדילה את גיל כל הפרטים באוכלוסיה בשנה. לכל סוג חיה יש את הגיל בו היא מגיעה לפורקן (ראו מטה) ורק כאשר הם עוברת את גיל זה, הן תהפוכנה לבוגרת – כלומר, כאלה שיכולות להעמיד צאצאים.

void printCommunity() const;	**(בשונה מתרגיל בית 2)**הדפסת הפרטים שיש באוכלוסייה ברגע זה (על ידי שימוש בprintAnimal).
void printCommunitySize() const;	הפונקציה מדפיסה את מספר הפרטים שיש ברגע זה באוכלוסיה, על ידי שימוש במחרוזת: "number of animals in the community is : “ ולאחר מכן מספר החיות
void ancestorsPath (const vector<string> &travelOrders) const;	המתודה מקבלת וקטור של מחרוזות כאשר בכניסה הראשונה בוקטור נמצא שם של חיה כלשהי מאוכלוסיה. לאחר מכן שאר הכניסות בוקטור פרט לאחרונה מכילות את המחרוזת “m” או "f "והכניסה האחרונה בוקטור מכילה את המחרוזת “end.” בשלב הראשון המתודה מדפיסה את הפרטים של החיה ששמה נמצא בכניסה הראשונה. בשלב הבא ע"פ הסדר של האותיות בוקטור אם מופיעה האות m אז מודפס הפרטים של האמא של החיה ואם מופיעה האות f אז הפרטים של האבא. לאחר מכך בהתאם לאות יודפסו הפרטים של הסבא או הסבתא וכך הלאה שאר האבות הקדמונים של החיה. במידה ומנסים לגשת לחיה שלא קיימת (למשל אם היא מתה) תודפס המחרוזת ”failed“ וה"טיול" ימשיך מהחיה האחרונה שהייתה תקינה. אם שם החיה שממנה מתחילים לסייר לא קיים באוכלוסיה תודפס המחרוזת "The animal was not found"
void makeSound(const string& name) const;	מתודה אשר מדפיסה את הרעש אשר החיה משמיעה כאשר קוראים לה (ראו טבלה מטה). name - שם החיה
void createNewAnimal(const string& name, const string& type, Gender gender);	מתודה אשר מייבאת חיה חדשה לגן החיות (ללא קשר לשושלות הקיימות בגן החיות.) name – שם החיה החדשה type – סוג החיה (ראו מטה) Gender – מין החיה

<u>דרישות מחלקה Animal:</u>

Method Name	Description
Animal(Animal* motherPtr, Animal* fatherPtr, const string&	בונה המחלקה. מאתחל את המאפיינים הבאים של החיה:

animalName, Gender animalGender, AnimalType type);	• motherPtr –פוינטר אל האמא של החיה, אם אין כזו יאותחל להיות NULL. • fatherPtr –פוינטר לאבא של החיה, אם אין כזה יאותחל להיות NULL. • animalName- שם החיה. • animalGender -מין החיה. • type - סוג החיה (ראו פירוט בDefinitions.h)
virtual ~Animal();	הורס את החיה.
virtual void printAnimal() const;	**(בשונה מתרגיל בית 2)** פונקציה אשר מדפיסה למסך את הפרטים של האובייקט לפי הפורמט (המודגשים הם הפרטים שעליכם להחליף) ***Type***: Name: ***name*,** Gender: ***gender***, Is adult: **bool,** Mother name: ***name***, Father name: ***name***, Number of children: ***numOfChildrens*** כאשר: **Type** – סוג החיה **name** – השם הרלוונטי (אם אין הורה, יש להחליף במחרוזת ריקה) **gender** – מין החיה ("male" או "female") **bool** – האם בוגר ("true" או "false") **numOfChilderns** – מספר ילדים של החיה
virtual void grow();	הפונקציה הופכת חיה לבוגרת, כך שיכולה להעמיד צאצאים **שימו לב – חיה יכולה להפוך לבוגרת רק כשהיא מגיעה לפרקה, ראו טבלה מטה**
virtual Animal* reproduce(Animal* partner, const string& descendantName, Gender decendantGender);	מתודת הרביה, מקבלת פוינטר לפרטנר פוטנציאלי, מחרוזת עם שם הצאצא ומין וצאצא. המתודה מחזירה פוינטר לצאצא אם התהליך הצליח ואחרת מחזירה פוינטר ל-NULL.
virtual void makeSound() const =0;	מתודה הפעלת קול, לפי פורמט ההדפסה ***SOUND*** כאשר sound מתאר את הצליל. (רמז – ניתן להשתמש בהגדרות אשר נמצאות בDefinitions.h הנתון)
virtual Animal* createNewAnimal(const string &descendantName, Gender decendantGender, Animal *mother, Animal *father) const = 0;	מתודה יצירת חיה חדשה (רמז – איננה קשורה למתודה בעלת השם הזהה בZoo).

ממשק המחלקות הנוספות:

Class name	Data members	member methods

Lion	static const string _SOUND;
	static const size_t LION_MATURE_AGE;
	static const string LION_SOUND;
		Lion(Animal* motherPtr, Animal* fatherPtr, const string& animalName, Gender animalGender);
		virtual void makeSound() const;
		virtual void grow();
		virtual void printAnimal() const;
		virtual Animal* reproduce(Animal *partner, const string &descendantName, Gender decendantGender);
		virtual Animal *createNewAnimal(const string &descendantName, Gender decendantGender, Animal *mother, Animal *father) const;
		virtual ~Lion();

Duck	static const string _SOUND;
	static const size_t _duckMatureAge;
	static const string DUCK_SOUND;
		Duck(Animal* motherPtr, Animal* fatherPtr, const string& animalName, Gender animalGender);
		virtual void makeSound() const;
		virtual Animal * createNewAnimal(const string &descendantName, Gender decendantGender, Animal *mother, Animal *father) const;
		virtual void grow();
		virtual void printAnimal() const;
		virtual Animal* reproduce(Animal *partner, const string &descendantName, Gender decendantGender);
		virtual ~Duck();

Proboscidea	static const string PROBOSCIDEA_SOUND;
	static const size_t _proboscideaMatureAge;
	static const string _SOUND;
		Proboscidea(Animal* motherPtr, Animal* fatherPtr, const string& animalName, Gender animalGender, AnimalType animalType);
		virtual void makeSound() const;
		virtual void grow();
		virtual void printAnimal() const = 0;
		virtual ~Proboscidea();

Mammoth		Mammoth(Animal* motherPtr, Animal* fatherPtr, const string& animalName, Gender animalGender);
		virtual void printAnimal() const;
		virtual Animal * reproduce(Animal *partner, const string &descendantName, Gender decendantGender);
		virtual Animal* createNewAnimal(const string &descendantName, Gender decendantGender, Animal *mother, Animal *father) const;
		virtual ~Mammoth();

Elephant		Elephant(Animal* motherPtr, Animal* fatherPtr, const string& animalName, Gender animalGender); virtual void printAnimal() const; virtual Animal * createNewAnimal(const string

		&descendantName, Gender decendantGender, Animal *mother, Animal *father) const;
		virtual Animal * reproduce(Animal *partner, const string &descendantName, Gender decendantGender);
		virtual ~Elephant();

- כל תכונות המחלקות פרטיות (All class members are private), גם הקיימים וגם החדשים (במידה ותצטרכו להוסיף כאלה) למעט למחלקה Animal בה התכונות הרלוונטיות לירושה (**ורק הן**) הינן protected.
- אין לשכפל מידע קיים ולשמור אותו תחת שם אחר.
- יש להדפיס את הפלט בדיוק כפי שניתן שמתואר בהמשך.

הקלט לתוכנה:

הסימולציה קולטת את נתוני הפקודות מערוץ הקלט הסטנדרטי (cin). הקלט מחולק לפקודות וכל פקודה תופסת שורה אחת בדיוק בקלט. סדר ביצוע הפקודות חייב להיות לפי סדר הופעתן בקלט (FIFO).

בטבלה הבאה, X מייצג משתנה (לדוגמא שם החיה, קצב אכילה וכו').

הסימולציה מתנהלת על ידיד הפקודות הבאות:

תיאור הפקודה	מבנה שורת הפקודה	פרמטרים
שורת אתחול התוכנית	X1 X2 Type	X1 – שם הזכר הראשון X2 – שם הנקבה הראשונה Type – סוג החיה של הזוג הראשון
רביה בין שני פרטים	animalReproduce X1 X2 X3 X4	X1 – שם ההורה הראשון X2 – שם ההורה השני X3 – שם הצאצא X4 – מין האצא
הרדמה של פרט	euthanize X	X – מחרוזת עם שם הפרט שהולך להיות מורדם
הגדלת גיל כל החיות בשנה	growUp	ללא
הדפסת פרטי האוכלוסייה	printCommunity	ללא
גודל האוכלוסיה	communitySize	ללא
טיול בעץ המשפחה של פרט מסויים.	ancestorsPath X Y1 Y2… Yk,”end”	X – מחרוזת עם שם הפרט שממנו מתחילים לטייל במסלול האבות הקדמונים. Y_i – פעולה שיש לעשות בתוך הסריקה יכולה לקבל את אחת מהמחרוזות הבאות : “m” או “f” בלבד.
ייבוא חיה לגן	add_animal X1 X2 X3	X1 – שם החיה X2 – סוג החיה

		X3 – מין החיה
השמעת קול	make_sound X1	X1 – שם החיה אשר אמורה לעשות צליל
יציאה מהתוכנית	quit	ללא

- ניתן להניח כי הקלט מכיל רק פקודות בפורמט הנ"ל

התנהגות החיות:

בתרגיל זה נגדיר 4 סוגים שונים של חיות:

Animal Type	**Mature Age**[1]	**Sound**
Lion	3	***RWAR***
Duck	1	***QUACK***
Mammoth	5[2]	***TRUMPET***
Elephant	5[2]	***TRUMPET***

- שימו לב איזה מחלקה לא נמצאת פה (זאת לא טעות)

הפלט של המערכת:

עבור הקלט:

```
ducky della Duck
growUp
printCommunity
animalReproduce ducky della Huey male
animalReproduce ducky della Dewey male
animalReproduce ducky della Louie male
add_animal Webby Duck female
printCommunity
make_sound Webby
quit
```

[1] הגיל בו החיה מגיעה לפרקה
[2] טכנית נכון, אבל לרוב בגיל 18 בפרקטיקה (נחסך מכם לצורך פשטות)

התוכנית תדפיס:

Duck: Name: ducky, Gender: male, Is adult: True, Mother name: , Father name: , Number of children: 0
Duck: Name: della, Gender: female, Is adult: True, Mother name: , Father name: , Number of children: 0
Duck: Name: ducky, Gender: male, Is adult: True, Mother name: , Father name: , Number of children: 3
Duck: Name: della, Gender: female, Is adult: True, Mother name: , Father name: , Number of children: 3
Duck: Name: Huey, Gender: male, Is adult: False, Mother name: della, Father name: ducky, Number of children: 0
Duck: Name: Dewey, Gender: male, Is adult: False, Mother name: della, Father name: ducky, Number of children: 0
Duck: Name: Louie, Gender: male, Is adult: False, Mother name: della, Father name: ducky, Number of children: 0
Duck: Name: Webby, Gender: female, Is adult: False, Mother name: , Father name: , Number of children: 0
QUACK

דרישות המימוש:

1. המימוש חייב להכיל לפחות את הקבצים הבאים:
   i. main.cpp – כפי שניתן על ידי צוות הקורס בקובץ המצורף.
   אין לשנות קובץ זה והמימוש שלכם חייב להשתמש בפונקציית main המוגדרת בו.
   ii. Definitions.h – קובץ אשר מסופק לכם על ידי צוות הקורס, על מנת להקל עליכם עם קבועים שונים. מותר להוסיף פונקציות ו/או קבועים (במקרה הצורך).
   iii. Zoo.h – מחלקה המייצגת את גן החיות. יש לממש כפי שמוגדר בטבלה מעלה, מותר להוסיף מתודות ו/או תכונות (במקרה הצורך).
   iv. Animal.h – מחלקה אבסטרקטית המייצגת חיה. יש לממש כפי שמוגדר בטבלה מעלה, מותר להוסיף מתודות ו/או תכונות (במקרה הצורך).
   v. Lion.h – מחלקה המייצגת אריה. יש לממש כפי שמוגדר בטבלה מעלה, מותר להוסיף מתודות ו/או תכונות (במקרה הצורך).
   vi. Duck.h – מחלקה המייצגת ברווז. יש לממש כפי שמוגדר בטבלה מעלה, מותר להוסיף מתודות ו/או תכונות (במקרה הצורך).
   vii. Proboscidea.h – מחלקה אבסטרקטית המייצגת בעלי חדק (פילאים). יש לממש כפי שמוגדר בטבלה מעלה, מותר להוסיף מתודות ו/או תכונות (במקרה הצורך).
   viii. Elephant.h – מחלקה המייצגת פיל. יש לממש כפי שמוגדר בטבלה מעלה, מותר להוסיף מתודות ו/או תכונות (במקרה הצורך).
   ix. Mammoth.h – מחלקה המייצגת ממותה. יש לממש כפי שמוגדר בטבלה מעלה, מותר להוסיף מתודות ו/או תכונות (במקרה הצורך).
   x. קבצי מימוש של המודולים הנ"ל (קבצי הcpp).
   xi. קבצי קוד נוספים, במידה ותמצאו לנכון.

2. עליכם לשמור על כללי תכנות נאות כפי שפורסם במסמך במודל, ובפרט להקפיד על:
   i. אתחול משתנים ולתת להם שמות משמעותיים.
   ii. לא לכתוב ליטרלים בקוד (אלא להגדיר קבועים מתאימים).
   iii. פרט לקיצורים וראשי התיבות המופיעים במסמך, ראשי תיבות **לא** נחשב לשם בעל משמעות

3. עליכם להשתמש בדוגמאות מההרצאות והתרגולים. כמו כן אתם רשאים להשתמש בפונק' נוספות של טיפוסי ספריה הנ"ל שלא למדנו (לפי תקן c++98 בלבד, כמובן).
   i. **אין לייבא ולהשתמש בספריות אשר לא נלמדו בכיתה**
4. עליכם לתעד את הקוד שלכם באנגלית בלבד (אין צורך לדאוג לרמת האנגלית). יש לתעד לפי סגנון התיעוד שמופיע במודל.
5. אין בתרגיל זה דרישה של יעילות זמן.
6. עבודה עם (בפרט על) מיכלים סדרתיים תעשה אך ורק בעזרת איטרטור המתאים ביותר
7. הקדישו תשומת לב ומחשבה לאופן בו הפונקציות שלכם מקבלות פרמטרים.
8. יש להקפיד **שאין** דליפות זיכרון.
9. יש להקפיד כי פלט הקבצים זהה לחלוטין לפלט הניתן לכם.

חלק יבש

ציירו את עץ הירושה שמתואר בתרגיל זה. ציינו במפורש את קשרי הHas a ו Is a.
הוסיפו הסבר קצר על עץ הירושה.

הוראות הגשה:

1. התרגיל להגשה בזוגות **בלבד**.
2. הקוד חייב להיכתב על פי מסכמות כתיבת הקוד (Coding Conventions) בקורס. קוד של עומד במסכמות לא יזכה במלוא הניקוד.
3. **חובה** לקמפל את התרגיל בעזרת סביבת TDM GCC שעליכם להגדיר על פי המדריך. תוכנית אשר תצליח להתקמפל בסביבה אחרת ולא בסביבה זו **תחשב כקוד שלא עובר קומפילציה**.
4. ההגשה חייבת להכיל קובץ ZIP יחיד בלבד (ולא קובץ RAR וכדומה) המכיל:
   - תיקייה בשם code ובה כל קבצי קוד המקור (h/cpp), ללא קבצי ההרצה (exe.)
   - קובץ pdf אשר מכיל את החלק היבש.
5. שם הקובץ חייב להיות hw3_xxxxxxxxx_yyyyyyyyy.zip כאשר xxxxxxxxx ו-yyyyyyyyy הם מספרי תעודות הזהות של המגישים כולל ספרת ביקורת.
6. ההגשה היא אלקטרונית בלבד, דרך אתר ה-moodle של הקורס. תרגילים שיוגשו בכל דרך אחרת לא ייבדקו.
7. אין להגיש את אותו הקובץ פעמיים. התרגיל יוגש על ידי **אחד** מבני הזוג.
8. תרגיל בית של יוגש על פי הוראות ההגשה – **לא ייבדק.(כלומר ציון 0)**
9. יש להקפיד על יושרת הכנת התרגיל וההגשה.
10. קוד אשר לא יעבור קומפילציה יזכה לציון 0.

בהצלחה!