# Допълнителна подготовка:Полета, свойства, методи и конструктори

## Пътувания с коли

Задачата ви е да напишете програма, която пази информация за автомобили, за това колко гориво имат и поддържа методи за движение на колите. Дефинирайте клас **Car** с информация за **модела, количеството гориво, разхода на гориво за 1 км.** и **пропътуваното разстояние**. Моделът на автомобилите е **уникален** - няма да има две коли с един и същи модел.

На първия ред на входните данни ще получите число **N** – броят на автомобилите, които ще следите. На всеки от следващите **N** реда ще има информация за по една кола в следния формат “<**Модел> <КоличествоГориво> <РазходНаГоривоЗа1км>**”. Всички **коли започват с пропътувани 0 км**.

След тези **N** реда, до достигане на команда “**End**”, ще получавате команди във следния формат “**Drive <МоделКола> <бройКм>**”. Реализирайте в класа **Car** метод, изчисляващ дали колата може да измине това разстояние или не. Ако да, **горивото на колата** трябва да бъде **намалено** с количеството на горивото, използвано за пътуването, а **изминатите от нея километри** трябва да бъдат **увеличени** с пропътуваните километри. Ако няма да може да го пропътува, колата не трябва да се движи (т.е. количеството на горивото и пропътуваните от нея километри трябва да останат същите), а на конзолата да се отпечата “**Insufficient fuel for the drive**”. След достигане на команда “**End**” трябва да се отпечата **всяка кола** и нейното **текущо количество гориво,** както и **пропътуваните километри във формата** ““**Модел> <КоличествоГориво> <пропътуваниКм>**”, където количеството гориво трябва да е отпечатано с **две цифри** след десетичния знак.

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 2AudiA4 23 0.3BMW-M2 45 0.42Drive BMW-M2 56Drive AudiA4 5Drive AudiA4 13End | AudiA4 17.60 18BMW-M2 21.48 56 |
| 3AudiA4 18 0.34BMW-M2 33 0.41Ferrari-488Spider 50 0.47Drive Ferrari-488Spider 97Drive Ferrari-488Spider 35Drive AudiA4 85Drive AudiA4 50End | Insufficient fuel for the driveInsufficient fuel for the driveAudiA4 1.00 50BMW-M2 33.00 0Ferrari-488Spider 4.41 97 |

## Застъпване на правоъгълници

Създайте клас **Rectangle**. Той трябва да съдържа полета **ID, широчина, височина** и **координатите на неговия горен ляв ъгъл** (**по хоризонтала и по вертикала**). Създайте **метод**,който **получава като параметър друг обект Rectangle**, проверява дали двата правоъгълника се **застъпват** и **връща true или false**.

На първия ред ще получите **броя правоъгълници** – **N** и броя **проверки за застъпване** – **M**. На следващите **N** ще получавате правоъгълници с тяхното **ID, широчина, височина и координати**. На последните **M** реда ще са **двойки от ID-та** на правоъгълници. Отпечатате дали при всяка от тези двойки **има** **застъпване.**

Винаги ще получавате валидни данни. Няма нужда да проверявате дали правоъгълниците съществуват.

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 2 1Pesho 2 2 0 0Gosho 2 2 0 0Pesho Gosho | true |

## Продавач на коли

Дефинирайте два класа **Car** и **Engine.** Класът за колите **Car** има полета за **модел, двигател, тегло** и **цвят** (**model, engine, weight**, **color**). Всеки двигател (**Engine**) има характеристики модел, мощност, кубатура и КПД (**model**, **power, displacement**, **efficiency**). **Теглото** и **цвета** на колата и **кубатурата** и **КПД-то** на двигателя ѝ са **незадължителни** данни.

На първия ред ще получите число **N**, показващо колко реда с информации за двигатели ще получите, на всеки от следващите **N** реда ще има информация за по един **двигател** в следния формат “<**Model> <Power> <Displacement> <Efficiency>**”. След редовете с двигателите, на следващия ред ще получите число **M** – указващо броя на колите, които следват. На всеки от следващите **M** реда ще има информация за една **кола** в следния формат “<**Model> <Engine> <Weight> <Color>**”, където двигателят ще е **модел на съществуващ (описан вече) двигател**. Когато създавате обект за **кола**, трябва да пазите **указател към** **точния двигател**, вместо само модела на двигателя. Обърнете внимание на това, че незадължителните характеристики **може да липсват** от форматираните данни.

Вашата задача е да отпечатате всяка кола (в реда, в който сте ги получили) и информацията за нея във вида, посочен по-долу; ако някое от незадължителните полета не е посочено, отпечатайте “**n/a**” на негово място:

**<CarModel>:
 <EngineModel>:
 Power: <EnginePower>
 Displacement: <EngineDisplacement>
 Efficiency: <EngineEfficiency>
 Weight: <CarWeight>
 Color: <CarColor>**

### Бонус\*

Предефинирайте методите ToString(), така че да имате многократно използваем начин за извеждане на тези обекти.

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 2V8-101 220 50V4-33 140 28 B3FordFocus V4-33 1300 SilverFordMustang V8-101VolkswagenGolf V4-33 Orange  | FordFocus: V4-33: Power: 140 Displacement: 28 Efficiency: B Weight: 1300 Color: SilverFordMustang: V8-101: Power: 220 Displacement: 50 Efficiency: n/a Weight: n/a Color: n/aVolkswagenGolf: V4-33: Power: 140 Displacement: 28 Efficiency: B Weight: n/a Color: Orange |
| 4DSL-10 280 BV7-55 200 35DSL-13 305 55 A+V7-54 190 30 D4FordMondeo DSL-13 PurpleVolkswagenPolo V7-54 1200 YellowVolkswagenPassat DSL-10 1375 BlueFordFusion DSL-13 | FordMondeo: DSL-13: Power: 305 Displacement: 55 Efficiency: A+ Weight: n/a Color: PurpleVolkswagenPolo: V7-54: Power: 190 Displacement: 30 Efficiency: D Weight: 1200 Color: YellowVolkswagenPassat: DSL-10: Power: 280 Displacement: n/a Efficiency: B Weight: 1375 Color: BlueFordFusion: DSL-13: Power: 305 Displacement: 55 Efficiency: A+ Weight: n/a Color: n/a |

## Треньор на покемони

Вие искате да сте най-добрия треньор на покемони, по-добър от всеки друг, така че сте се заели да ловите покемони. Дефинирайте клас **Trainer** и клас **Pokemon**. Треньорът трябва да има **име**, **брой значки** и **колекция от покемони. Покемонът** има **име, елемент** и **здраве**, всички стойности са **задължителни**. Всеки треньор **започва с 0 значки**.

От конзолата ще получите неизвестно колко редове, след които ще следва команда “**Tournament**”. Всеки от тези редове ще носи информация за покемона и треньора, който го е хванал във формата “<**ИмеНаТреньор> <ИмеНаПокемон> <ЕлементНаПокемона> <ЗдравеНаПокемона>”** където **ИмеНаТреньор** е името на треньора, хванал покемона; имената са **уникални,** няма как да има двама треньори с еднакви имена. След получаване на команда “**Tournament**” неизвестен брой редове ще съдържат като команда един от трите елемента “**Fire**”, “**Water**”, “**Electricity**”, които продължават, докато се получи команда “**End**”. За всяка от тези команди трябва да проверите дали треньорът има поне един покемон с дадения елемент. Ако да, треньорът получава 1 значка, в противен случай всичките му покемони **губят 10 точки здраве**, а ако даден покемон падне до **0 или по-малко точки здраве той умира** и трябва да бъде изтрит от колекцията на треньора. След като бъде получена команда “**End**” трябва да отпечатите всички треньори, **сортирани според броя на значките, които имат, в намаляващ ред** (ако двама треньори имат еднакъв брой значки те трябва да са сортирани според реда на тяхното появяване във входните данни),във формата “<**ИмеНаТреньор> <Значки> <БройПокемони>**”.

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| Pesho Charizard Fire 100Gosho Squirtle Water 38Pesho Pikachu Electricity 10TournamentFireElectricityEnd | Pesho 2 2Gosho 0 1 |
| Stamat Blastoise Water 18Nasko Pikachu Electricity 22Jicata Kadabra Psychic 90TournamentFireElectricityFireEnd | Nasko 1 1Stamat 0 0Jicata 0 1 |

## Google

Google винаги ви наблюдава, така че не би трябвало да сте изненадани, че те знаят всичко за вас (дори и за вашата покемон колекция). И понеже вие сте наистина добри в писането на класове от Google са ви помолили да напишете клас, който съдържа цялата информация, която те искат да събират за хората.

От конзолата ще получите неясно колко редове, завършващи накрая с команда “**End**”. На всеки от тези редове ще е информацията за един човек в един от следните формати:

* “**<Име> company <имеНаФирма> <отдел> <заплата>**”
* “**<Име> pokemon <имеНаПокемон> <типНаПокемона>”**
* “**<Име> parents <имеНаРодител> <рожденДенНаРодителя>**”
* “**<Име> children <имеНаДете> <рожденДенНаДете>**”
* “**<Име> car <моделНаКолата> <скоростНаКолата>**”

Вие трябва да структурирате цялата информация за всеки човек в клас с вложени класове. Имената на хората са **уникални** - няма двама души с еднакви имена, човек може да има **само 1 фирма** и **кола**, но има **множество родители, деца** и **покемони**. След като се получи команда “**End**” на следващия ред ще получите едно име и трябва да отпечатите цялата информация за този човек. Имайте в предвид, че информацията може да се промени във входните данни, например ако се получат множество редове, които указват фирмата на човека, само **последния** е този, който трябва да бъде запомнен. Заплатата трябва да бъде изведена с **два знака** след десетичния разделител.

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| PeshoPeshev company PeshInc Management 1000.00TonchoTonchev car Trabant 30PeshoPeshev pokemon Pikachu ElectricityPeshoPeshev parents PoshoPeshev 22/02/1920TonchoTonchev pokemon Electrode ElectricityEndTonchoTonchev | TonchoTonchevCompany:Car:Trabant 30Pokemon:Electrode ElectricityParents:Children: |
| JelioJelev pokemon Onyx RockJelioJelev parents JeleJelev 13/03/1933GoshoGoshev pokemon Moltres FireJelioJelev company JeleInc Jelior 777.77JelioJelev children PudingJelev 01/01/2001StamatStamatov pokemon Blastoise WaterJelioJelev car AudiA4 180JelioJelev pokemon Charizard FireEndJelioJelev | JelioJelevCompany:JeleInc Jelior 777.77Car:AudiA4 180Pokemon:Onyx RockCharizard FireParents:JeleJelev 13/03/1933Children:PudingJelev 01/01/2001 |

### Бонус\*

Предефинирайте метода ToString() в дефинираните класове, за да стандартизирате извеждането на обектите.

## Родословно дърво

Решили сте да направите родословно дърво, така че сте поразпитали баба си за фамилията. За съжаление тя помни само откъслечна информация за предците ви, затова на вас се пада честта да обобщите информацията и да построите родословното дърво.

На първия ред на входните данни ще получите или име, или дата на раждане във формати “<**Име> <Фамилия>”** или **“ден/месец/година**”. Вашата задача ще бъде да откриете информацията за човека в родословното дърво. На следващите редове до команда “**End**” ще получавате информация за вашите предци, която ще ви е нужна за построяване на фамилното дърво.

Информацията ще бъде в един от следните формати:

* “**Име Фамилия - Име Фамилия**”
* “**Име Фамилия - ден/месец/година**”
* “**ден/месец/година - Име Фамилия**”
* “**ден/месец/година - ден/месец/година**”
* “**Име Фамилия ден/месец/година**”

Първите 4 формата разкриват семейна връзка – **лицето отляво е родител на лицето отдясно** (както виждате, не е задължително форматът да съдържа имена, например 4-тия формат означава, че лицето, родено на датата отляво е родител на лицето, родено на датата отдясно). Последният формат свързва друг тип информация - например **лицето с това и това име е родено на тази и тази дата**. **Имената** и **рожденните дати** са **уникални** – няма да има двама души със съвпадащо име или рожденна дата, **винаги** ще има достатъчно данни за да се състави родословното дърво (имената и рожденните дати на всички хора са известни и всеки от тях ще има поне една връзка с някой друг от родословното дърво).

След получаването на команда “**End**” трябва да отпечатите цялата информация за лицето, чието име или рожденна дата сте получили на първия ред – неговото **име, рожден ден, родители и деца** (проверете примерите за изисквания формат). Хората в списъка на родителите и децата трябва да бъдат подредени според тяхното първо появяване във входните данни (без значение дали са били подадени като рожденна дата или като име - например в първата серия примерни данни Стамат е преди Пенка, защото той е споменат пръв на втория ред, докато тя се появява за пръв път на третия).

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| Pesho Peshev11/11/1951 - 23/5/1980Penka Pesheva - 23/5/1980Penka Pesheva 9/2/1953Pesho Peshev - Gancho PeshevGancho Peshev 1/1/2005Stamat Peshev 11/11/1951Pesho Peshev 23/5/1980End | Pesho Peshev 23/5/1980Parents:Stamat Peshev 11/11/1951Penka Pesheva 9/2/1953Children:Gancho Peshev 1/1/2005 |
| 13/12/199325/3/1934 - 4/4/1961Poncho Tonchev 25/3/19344/4/1961 - Moncho TonchevToncho Tonchev - Lomcho TonchevMoncho Tonchev 13/12/1993Lomcho Tonchev 7/7/1995Toncho Tonchev 4/4/1961End | Moncho Tonchev 13/12/1993Parents:Toncho Tonchev 4/4/1961Children: |

## Министерство на образованието и науката (МОН)

* Настоящият курс (презентации, примери, задачи, упражнения и др.) е разработен за нуждите на Национална програма "**Обучение за ИТ кариера**" на МОН за подготовка по професия "Приложен програмист".

* Курсът е базиран на учебно съдържание и методика, предоставени от **фондация "Софтуерен университет"** и се разпространява под **свободен** **лиценз CC-BY-NC-SA** (Creative Commons Attribution-Non-Commercial-Share-Alike 4.0 International).

