СОФИЙСКИ УНИВЕРСИТЕТ „СВ. КЛИМЕНТ ОХРИДСКИ“

ФАКУЛТЕТ ПО МАТЕМАТИКА И ИНФОРМАТИКА

ДЪРЖАВЕН ИЗПИТ

ЗА ПОЛУЧАВАНЕ НА ОКС “БАКАЛАВЪР ПО ИНФОРМАЦИОННИ СИСТЕМИ”

ЧАСТ I (ПРАКТИЧЕСКИ ЗАДАЧИ)

11.9.2014 г.

Моля, не пишете в тази таблица!			
Зад. 1		Зад. 5	
Зад. 2		Зад. 6	
Зад. 3		Зад. 7	
Зад. 4		Зад. 8	
Крайна оценка:			

Драги абсолвенти:

- Попълнете факултетния си номер в горния десен ъгъл на всички листа;
- Пишете само на предоставените листове без да ги разкопчавате;
- Ако имате нужда от допълнителен лист, можете да поискате от квесторите;
- Допълнителните листа трябва да се номерират, като номерата продължават тези от настоящия комплект;
- Всеки от допълнителните листа трябва да се надпише най-отгоре с вашите три имена и факултетен номер.
- **Решението на една задача трябва да бъде на същия лист, на който е и нейното условие (т.е. може да пишете отпред и отзад на листа със задачата, но не и на лист на друга задача).**
- Ако решението на задачата не се побира в един лист, трябва да поискате нов бял лист от квесторите. В такъв случай отново трябва да започнете своето решение на листа с условието на задачата и в края му да напишете „Продължава на лист № X”, където X е номерът на допълнителния лист, на който е вашето решение.
- Черновите трябва да бъдат маркирани, като най-отгоре на листа напишете „ЧЕРНОВА“.
- На един лист не може да има едновременно и чернова и белова.
- Времето за работа по изпита е 3 часа

Изпитната комисия ви пожелава успешна работа!

Задача 1. (10 т.) *Задачата да се реши на езика C++ или Java. В началото на вашето решение посочете кой език сте избрали*.

Дадени са координатите на N-точки, които са записани в масивите `float x[N], y[N]` по следния начин: координатите на `i`-тата точка са `(x[i], y[i])`.

Напишете функция `square`, която получава като аргументи броя на точките N и два масива `X` и `Y` съдържащи координатите им и извежда на екрана координатите на центъра и страната на най-малкия квадрат със страни успоредни на координатните оси, който обхваща всички дадени точки (всички дадени точки са във вътрешността му или на страните му).

Ако решавате задачата на Java, достатъчно е да напишете статична функция, която решава задачата.

Задача 2. (10 т.) *Задачата да се реши на езика C++ или Java. В началото на вашето решение посочете кой език сте избрали.*

Нека GameBoard е предварително дефинирана квадратна матрица от цели числа с размери N x N, представяща игрова дъска. Всеки елемент в матрицата има стойност 0 („земя”), 1 („огън”) или 2 („вода”). За две позиции в матрицата (i, j) и (i', j') казваме, че са съседни, ако $|i - i'| \leq 1$ и $|j - j'| \leq 1$.

А) Да се дефинира структура `Point`, описваща позиция на игровата дъска. Да се дефинира абстрактен клас (или интерфейс) `GamePlayer`, който описва играч на игровата дъска със следните операции:

- `getPosition`() – Връща позицията на играча на дъската;
- `allowedMoves`() – Връща списък (колекция) с всички възможни позиции, до които играчът може да достигне с един ход.

Б-1) Да се дефинира клас `Knight`, наследник на `GamePlayer`, описващ „сухопътен рицар“. Рицарят може да се придвижва само в такава съседна позиция, която е „земя” и не е в съседство с „огън”. Пример за достижими позиции за рицаря `K` е показан на диаграмата вдясно.

1	0	1
0	K	2
0	**0**	2

Б-2) Да се дефинира клас `SeaMonster`, наследник на `GamePlayer`, описващ „морско чудовище“. Морското чудовище може да се придвижва с произволен брой позиции по хоризонтала или по вертикала, но само по „вода”. Пример за достижими позиции за чудовището `S` е показан на диаграмата вдясно.

1	1	0	2	0
0	**2**	1	0	2
2	S	**2**	**2**	1
1	1	2	2	0
2	2	1	1	1

В) „Война“ наричаме такава подредба на играчите по дъската, при която на някоя от съседните позиции на всеки играч има друг играч. Да се дефинира функция

```
allMoves ([подходящ тип] players[, …])
```

която по даден списък (колекция) `players`, съдържащ произволен брой разнородни играчи, извежда на стандартния изход всеки възможен ход на играч от `players` такъв, че след изпълнението му списъкът с играчи да описва война. Информацията за ходовете да съдържа типа на играча, старата позиция и новата позиция.

Пример:

```
Knight (0,0) -> (1,1)
SeaMonster (2,2) -> (5,2)
```

Забележка: реализирайте всички конструктори и други операции, които смятате, че са необходими на съответните класове.

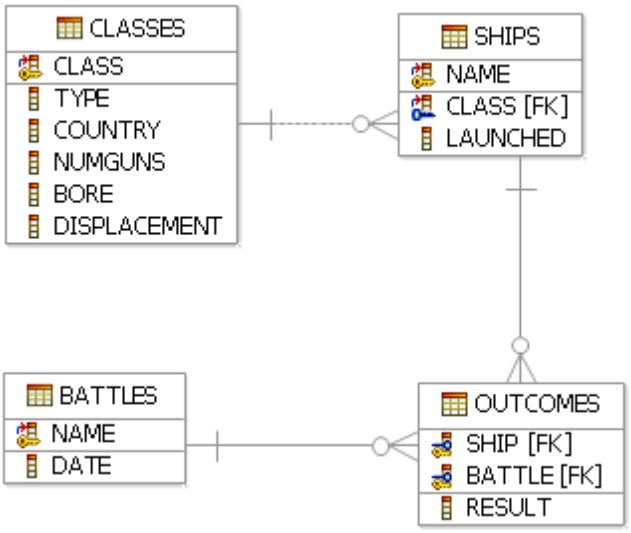
Задача 3. (10 т.) *Задачата да се реши на езика Scheme или Haskell. В началото на вашето решение посочете кой език сте избрали*.

Нека е даден списък L, който може да съдържа елементи от произволен тип. Напишете функция `permutations`, която получава такъв списък и връща списък с всички пермутации (възможни пренареждания) на неговите елементи. Резултатът да се върне като списък от списъци, в който всеки подсписък представя една пермутация на елементите на L.

Пример (Scheme):
`(permutations '(1 2 3))` → `( (1 2 3) (1 3 2) (2 1 3) (2 3 1) (3 1 2) (3 2 1) )`

Пример (Haskell):
`permutations [1,2,3]` → `[ [1,2,3], [1,3,2], [2,1,3], [2,3,1], [3,1,2], [3,2,1] ]`

Задача 4. (10 т.) Дадена е базата от данни Ships, в която се съхранява информация за кораби (*Ships*) и тяхното участие в битки (*Battles*) по време на Втората световна война. Всеки кораб е построен по определен стереотип, определящ класа на кораба (*Classes*).



Таблицата ***Classes*** съдържа информация за класовете кораби:

- *class* – име на класа, първичен ключ;
- *type* – тип (‘bb’ за бойни кораби и ‘bc’ за бойни крайцери);
- *country* – държавата, която строи такива кораби;
- *numGuns* – броят на основните оръдия;
- *bore* – калибърът им (диаметърът на отвора на оръдието в инчове);
- *displacement* – водоизместимост (в тонове).

Таблицата ***Ships*** съдържа информация за корабите:

- *name* – име на кораб, първичен ключ;
- *class* – име на неговия клас, външен ключ към Classes.class;
- *launched* – годината, в която корабът е пуснат на вода.

Таблицата ***Battles*** съхранява информация за битките:

- *name* – име на битката, първичен ключ;
- *date* – дата на провеждане.

Таблицата ***Outcomes*** съдържа информация за резултата от участието на даден кораб в дадена битка (колоните ship и battle заедно формират първичния ключ):

- *ship* – име на кораба, външен ключ към Ships.name;
- *battle* – име на битката, външен ключ към Battle.name;
- *result* – резултат (потънал-‘sunk’, повреден – ‘damaged’, победил – ‘ok’).

За така описаната база данни, решете следните задачи:

1. Напишете заявка, която извежда имената на всички кораби, пуснати на вода в година, в която е имало битка (не е задължително корабът да е участвал в нея).

2. Напишете заявка, която за всички държави, които имат най-много 3 (евентуално 0) кораба, извежда името на държавата и броя потънали кораби (който също може да бъде 0).

Задача 5. (10 т.) Моделирайте база от данни за Обиколката на Франция (le Tour de France) през 2014 г. Базата от данни трябва да съдържа следната информация:

Отбори (Teams)

- Име на отбор (tname) - уникален идентификатор за всеки отбор, низ до 20 символа
- Държава (tcountry), за която се състезава отбора, низ точно 3 символа
- Брой победи на предходни състезания le Tour de France (num_tf), цяло число
- Брой етапни победи на предходни състезания le Tour de France (num_stf), цяло число
- Брой спечелени жълти фланелки на предходни състезания le Tour de France (num_yj), цяло число

Колоездачи (Riders)

- Име на колоездач (rname) – уникален идентификатор за всеки колоездач , низ
- Номер на фланелка (rnum), цяло число
- Дата на раждане (birthdate) - дата
- Височина на колоездач (height) - цяло число
- Килограми на колоездач (weight) - цяло число
- Държава (rcountry) от която е колоездача - низ точно 3 символа
- Град (rcity) от който е колоездач - низ до 20 символа

Етапи (Stages)

- Номер на етап (snumber) – уникален идентификатор за всеки етап, число
- Дата на провеждане на етапа (sdate) – дата
- Километри на етапа (km) - число
- Град начало на етапа (scity) - низ до 30 символа
- Град край на етапа (ecity) - низ до 30 символа

В сила са следните ограничения:

- Към един отбор може да участват много колоездачи (до 9), а един колоездач може да участва само в един отбор.
- В един етап могат да участват много колоездачи и един колоездач може да участва в много етапи на състезанието. За всяко участие на колоездач в етап, трябва да се пази и информация за мястото на което се е класирал (цяло число), за спечелени точки от етапа (цяло число), време за което е завършил етапа (от тип време), дали е спечелил бяла фланелка (цяло число - 0 или 1), дали е спечелил жълта фланелка (цяло число - 0 или 1) и дали е спечелил зелена фланелка (цяло число - 0 или 1).

Направете E/R модел на описаната по-горе база от данни. Отбележете първичните ключове. Преобразувайте направения от вас E/R модел в релационен модел.

Задача 6. (10 т.) Прочетете описанието на информационната система от задача 9. След това решете дадените по-долу задачи.

А) Да се изготви потребителски случай (пълно описание), който описва как даден студент може да използва системата, за да получи уверение, че учи във ФМИ, което да послужи пред някаква организация. Системата трябва да позволява на студента да изпрати заявка за издаване на уверение и след това да следи какъв е нейният статус (напр. получена, чакаща, обработва се и т.н.). Предполага се, че самото уверение ще се получава в хартиен вид и подпечатано във ФМИ. Потребителският случай да описва и какви проверки за коректност се правят при обработката (напр. дали лицето има студентски права, в коя специалност учи и т.н.).

Б) В потребителския случай включете поне два алтернативни сценария.

В) Напишете нефункционални изисквания към потребителския случай.

Г) Начертайте диаграма на последователност (Sequence), която описва последователността от стъпки в описания от вас потребителски случай. Диаграмата да включва подходящи етикети върху отделните елементи (напр. върху стрелките). Ако е нужно, напишете кратки разяснения към диаграмата.

Задача 7. (10 т.) Да се направи информационен модел на електронна автобиография на хора, който да замести хартиения документ. Да се аргументира изборът за всеки обект от модела, структурата на модела и броя на появяванията на обектите. Автобиографията съдържа: информация за имената, дата на раждане, месторождение и гражданство, информация за адреса (държава, град, улица, номер на улица), телефон и email, информация за образователните институции, които е завършил, специалностите, по които се е обучавал, вида образование, което е завършил и година на дипломиране. Това съдържание е записано в XML документ, в който има информация за нула, един или повече души. Да се състави описание на документния му тип с използване на DTD или XML Schema. Да се създадат примерен XML документ, в който има данни за поне двама души, и съответен DTD или XML Schema документ

Задача 8. (10 т.) „221“ е кодовото име на проект на информационна система за студенти във ФМИ. Системата трябва да предоставя профил на всеки студент, в който да може да се получава информация за учебната програма и разписанието за семестъра, както и текущи обяви и новини. Системата трябва да позволява размяна на съобщения между потребители и заявка за издаване на документ от администрацията. „221“ също трябва да дава достъп до текущата академична справка на студента и да предоставя връзка до електронни материали за всеки текущо провеждан курс. Системата трябва да бъде достъпна и през най-популярните мобилни устройства (смартфони и таблети) чрез специално оптимизиран интерфейс. „221“ трябва да се интегрира с базата от данни на ФМИ, съдържаща потребителски профили на студентите и информация за провежданите дисциплини. Трябва да бъде предоставен и административен интерфейс за преподаватели, чрез който да бъдат публикувани обяви и материали за дисциплините.

Да се предложи примерна структура на работните пакети (work breakdown structure) за проекта, с не повече от 4 нива и не повече от 20 възела.

Ч Е Р Н О В А

Ч Е Р Н О В А

ЧЕРНОВА