СОФИЙСКИ УНИВЕРСИТЕТ
„СВ. КЛИМЕНТ ОХРИДСКИ“

ФАКУЛТЕТ ПО МАТЕМАТИКА
И ИНФОРМАТИКА

ДЪРЖАВЕН ИЗПИТ
ЗА ПОЛУЧАВАНЕ НА ОКС “БАКАЛАВЪР ПО ИНФОРМАЦИОННИ СИСТЕМИ”

ЧАСТ I (ПРАКТИЧЕСКИ ЗАДАЧИ)
10.09.2018 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 вашия 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. *Задачата да се реши на един от езиците C или C++.*

Професор Х забравя постоянно паролата за своя таен сейф, която представлява последователност от цели числа. За да не е нужно да я помни, той скрива подсказки в кабинета си. На всеки ред от библиотеката му са подредени книги, най-много 20 реда книги, с до 30 книги на ред. Книгите на някои от редовете са подредени по азбучен ред на заглавията си, всяко от които е до 100 символа.

Паролата за сейфа на професор Х се определя от числата, които задават последователните дължини на думите в заглавията на книгите, разположени точно в средата на редовете, в които книгите са подредени в азбучен ред. Ако на реда има четен брой книги, за паролата се използва книгата, намираща се по-близо до началото на реда. Думите в заглавията на книгите са разделени от точно един интервал. Дължините на думите формират паролата в реда, в който се срещат, от най-горния към най-долния ред на библиотеката.
Библиотеката на Х може да се представи като двумерен масив **a** от низове с **m** реда по **n** низа всеки, представящи заглавията на книгите. Да се дефинира функция **`revealPassword`**, която по подадени **a**, **m** и **n**, извежда на стандартния изход паролата на професор Х като последователност от числа, разделени с по един интервал. Да се демонстрира употребата на функцията **`revealPassword`** в кратка програма.

Пример:
Нека библиотеката на професор Х се състои от 3 реда с по 3 книги всеки както следва:

Алгебра	Аналитична геометрия	Математически анализ
Увод в програмирането	Обектно-ориентирано програмиране	Структури от данни и програмиране
Бази от данни	Изкуствен интелект	Функционално програмиране

Тогава паролата на професор Х е поредицата от числа 10 9 9 8, получена от дължините на подчертаните думи. Думите “Обектно-ориентирано програмиране” не участват в паролата, защото книгите на втория ред не са подредени по азбучен ред на заглавията си.

Задача 2. *Задачата да се реши на езика C++.*

А) Разгледайте дадената по-долу програма. Да се довърши класът `Counter` така, че член-функцията `GetObjCount` да връща колко негови обекта съществуват в момента, в който тя бъде извикана. В решението НЕ МОЖЕ да се използват глобални променливи, глобални функции, други класове и т.н. То трябва да засяга само `Counter`.

```
#include <iostream>

class Counter {


public:
   Counter()
   {

   }

   ~Counter()
   {

   }

   Counter(const Counter& other)
   {

   }

   Counter& operator=(const Counter& other)
   {

   }

   static int GetObjCount()
   {

   }
};


int main()
{  // До всеки ред е указано какво ще се изведе
   std::cout << Counter::GetObjCount(); //0
   Counter* p = new Counter[10];
   std::cout << Counter::GetObjCount(); //10
   delete[] p;
   std::cout << Counter::GetObjCount(); //0
   Counter a, b;
   Counter c = a;
   a = b;
   std::cout << Counter::GetObjCount(); //3
   return 0;
}
```

Б) Нека класът `Derived` наследява `Counter` по следния начин:

```
class Derived : public Counter { };
```

Разгледайте дадения по-долу фрагмент. Под редовете извеждащи текст има коментари *"горният ред ще изведе"*. Срещу всеки от тях посочете какво ще се изведе. След фрагмента обяснете защо смятате така (обосновете отговора си). Решение, в което не е дадено обяснение се оценява с нула точки.

```
Derived d1;
std::cout << Derived::GetObjCount();
            // горният ред ще изведе:

Counter c;
std::cout << Counter::GetObjCount();
            // горният ред ще изведе:

Counter* p = new Derived;
std::cout << Counter::GetObjCount();
            // горният ред ще изведе:

std::cout << Derived::GetObjCount();
            // горният ред ще изведе:

delete p;
std::cout << Counter::GetObjCount();
            // горният ред ще изведе:

std::cout << Derived::GetObjCount();
            // горният ред ще изведе:
```

Тук обосновете отговора си:

Задача 3. *Задачата да се реши на един от езиците Scheme или Haskell. По-долу оградете името на езика, който сте избрали за вашето решение.*

Даден е списък от списъци от числа **ll** и числова функция **f**. Числото **x** наричаме "корен" на **f**, ако **f(x) = 0**. Да се попълнят по подходящ начин празните полета по-долу, така че функцията **sumMaxRoots** да намира сумата на корените на **f** в този списък от **ll**, в който **f** има най-много корени. Ако има няколко такива списъка, функцията да връща сумата на корените в първия по ред списък. Ако функцията няма корен сред числата в списъците на **ll**, функцията да връща **0**.
Упътване: можете да използвате наготово функциите `apply`, `filter`, `foldr`, `length`, `map`, `max`, `maximum`, както и всички стандартни функции в R^{5}RS за Scheme и Prelude за Haskell.

Scheme

```
(define (selectList l1 l2)
  (if _____________________________________ l1 l2))

(define (sumMaxRoots f ll)
  (_________
     (_________ selectList _________
          (_________ (lambda (l)
                (_________ (lambda (x) ___________________) l)) ll))))
```

Пример:
`(sumMaxRoots (lambda (x) (- (* x x x) x)) '((1 2 3) (-1 0 5) (1 4 -1))) → -1`

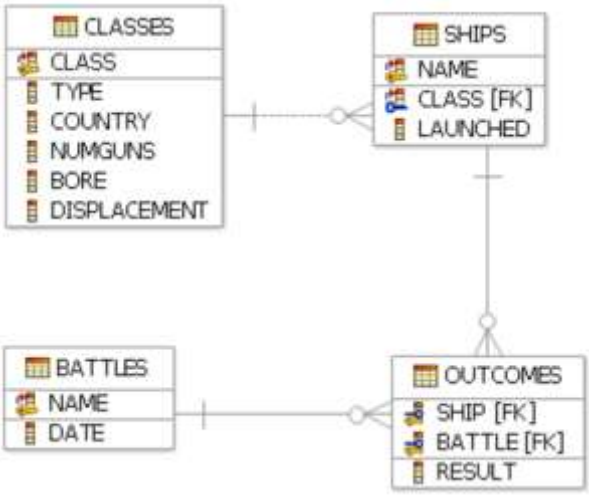
Haskell

```
selectList l1 l2 = if _____________________________________ then l1 else l2

sumMaxRoots f ll =
  _________
     (_________ selectList _________
        (_________ (\l -> [ _________ | x <- l, ___________________ ]) ll))
```

Пример:
`sumMaxRoots (\x -> x^3 - x) [ [1, 2, 3], [-1, 0, 5], [1, 4, -1] ] → -1`

Задача 4. Дадена е базата от данни `Ships`, в която се съхранява информация за кораби и тяхното участие в битки по време на Втората световна война. Всеки кораб е построен по определен стереотип, определящ класа на кораба. Таблицата `Classes` съдържа информация за класовете кораби:



- `class` – име на клас, първичен ключ;
- `type` – тип (`'bb'` за бойни кораби, `'bc'` за бойни крайцери);
- `country` – държава, която строи такива кораби;
- `numguns` – брой оръдия, може да приема стойност `null`;
- `bore` – калибър на оръдието (в инчове), може да приема стойност `null`;
- `displacement` – водоизместимост (в тонове), може да приема стойност `null`.

Таблицата `Ships` съдържа информация за корабите:

- `name` – име на кораб, първичен ключ;
- `class` – име на клас, външен ключ към `Classes.class`;
- `launched` – година, в която корабът е пуснат на вода, може да приема стойност `null`.

Таблицата `Battles` съхранява информация за битките:

- `name` – име на битка, първичен ключ;
- `date` – дата на провеждане.

Таблицата `Outcomes` съдържа информация за резултата от участието на даден кораб в дадена битка. Атрибутите `ship` и `battle` заедно формират първичния ключ.

- `ship` – име на кораб, външен ключ към `Ships.name`;
- `battle` – име на битка, външен ключ към `Battles.name`;
- `result` – резултат (потънал – `'sunk'`, повреден – `'damaged'`, победил – `'ok'`).

Забележка за всички таблици: За всички атрибути, за които не е указано, че могат да приемат стойност `null`, да се счита, че съществува `not null` ограничение.

1. Да се напише заявка, която извежда име на клас, годината на първата битка, в която кораб на този клас е участвал, годината на последната битка, в която кораб на този клас е участвал, и броя на всички различни битки, в които кораби на този клас са участвали, само за тези класове, започващи с буквата `N`. Ако за даден клас няма кораб, който да е участвал в битка, за съответните години да се върне стойност `null`.

2. Да се напише заявка, която да изведе имената на тези битки, за които броят на корабите от тип `'bb'`, участвали в тази битка, е по-голям от броя на корабите от тип `'bc'`, участвали в същата битка. Битки, в които не е участвал нито един кораб, да не се извеждат в резултата.

Задача 5. Информационна система съхранява информация за ТВ сериали. Съхранява се информация за сезони с данни за номер на сезон, име на продукция, начална дата на сезона и крайна дата на сезона. Съхранява се също и информация за епизоди с данни за номер на епизод, дата на излъчване на епизода и продължителност на епизода. Номерът на сезона и името на продукцията заедно определят уникално всеки сезон. Номерът на епизода е уникален в рамките на сезона. В информационната система се съхранява и информация за герои с данни за номер на герой — цяло положително число, уникално за всеки герой, име на герой, роля на герой и име на актьора, който играе или озвучава образа на героя. Един герой може да участва в много епизоди и в един епизод могат да участват много герои.

Задание:

а) Да се направи E/R модел на БД, която съхранява гореописаната информация. Да се начертае E/R диаграма на модела.

б) Да се преобразува E/R диаграмата към релационни схеми. Да се премахнат излишествата, където това е възможно.

в) Да се напише DDL код, съответстващ на релационните схеми. Да се реализират всички описани ограничения.

Задача 6. В контекста на системата за сериали, описана в Задача 5, разглеждаме процес, при който потребител избира епизод от сериал, който да гледа, въз основа на интерактивно попълване на анкета, предоставена му от системата. В анкетата се вземат предвид предишни взаимодействия с потребителя и се задават въпроси, целящи да определят какво точно би искал да гледа той. Например, такива въпроси могат да касаят жанра на сериала; участващи актьори; дали иска да гледа повторно даден епизод или да продължи гледането на започнат сезон и т. н.

Да се опише в пълен формат потребителски случай за процеса по препоръчване на епизод. Да се включат поне един алтернативен и един неуспешен сценарий. В описанието да се включат като минимум: актьори, които участват; предусловия; следусловия (резултати); взаимодействие с други потребителски случаи (ако е приложимо, но само с препратка). Да се посочат допълнителни нефункционални изисквания към този случай.

Задача 7. В контекста на системата и сценария, описани в задачи 5 и 6, начертайте UML диаграма на дейността (Activity Diagram), описваща процеса по препоръчане на епизод. Добавете кратък текст, поясняващ предположенията, които сте направили за системата. Добавете кратко текстово описание на диаграмата.

Задача 8. Пресметнете интеграла $\int_0^1 x^2 \operatorname{arctg} x \, dx$.

Чернова