СОФИЙСКИ УНИВЕРСИТЕТ
“СВ. КЛИМЕНТ ОХРИДСКИ”

ФАКУЛТЕТ ПО МАТЕМАТИКА
И ИНФОРМАТИКА

ДЪРЖАВЕН ИЗПИТ

ЗА ПОЛУЧАВАНЕ НА ОКС “БАКАЛАВЪР ПО ИНФОРМАЦИОННИ СИСТЕМИ”

ЧАСТ I (ПРАКТИЧЕСКИ ЗАДАЧИ)

Драги абсолвенти:

- Попълнете факултетния си номер в горния десен ъгъл на всички листове.
- Пишете само на предоставените листове, без да ги разкопчавате.
- 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 имате нужда от допълнителен лист, можете да поискате от квесторите.
- На един лист не може да има едновременно и чернова, и белова.
- Черновите трябва да се маркират, като най-отгоре на листа напишете “ЧЕРНОВА”.
- Ако решението на една задача не се побира на нейния лист, трябва да поискате нов бял лист от квесторите. Той трябва да се защипе с телбод към листа със задачата.
- Всеки от допълнителните листове (белова или чернова) трябва да се надпише най-отгоре с вашия факултетен номер.
- Черновите също се предават и се защипват в края на работата.
- Времето за работа по изпита е 3 часа.

Изпитната комисия ви пожелава успешна работа!

Задача 1. Задачата да се реши на езика C++.

1) Нека е дефиниран масивът
`int arr[] = { 1, 2, 3 };`
Срещу всеки от изразите да се посочи каква ще бъде неговата оценка.

```
arr[1] == *(arr+2) _____________________
arr == &arr[0]     _____________________
(arr+1) == &arr[1] _____________________
*arr == arr[0]     _____________________
```

2) Нека е дадена следната дефиниция:

```
void mystery(const char* str)
{
  while (*str && *(str+1)) {
    std::cout << *str;
    str += 2;
  }
}
```

Да се посочи какво ще изведе на екрана обръщението:

```
mystery("abcdef");
```

_____

3) Нека са дадени следните дефиниции:

```
char s1[] = "Hello";
char s2[] = "world!";
char result[80];
```

Да се довърши програмният фрагмент, така че след изпълнението му в `result` да се съхрани коректното представяне на низа `"Hello world!"`. На празните места трябва да се попълнят имената на подходящи стандартни функции за работа с низове.

```
_____________________(result, s1);
_____________________(result, " ");
_____________________(result, s2);
```

4) Да се довърши кодът на рекурсивните функции, така че `f` да проверява дали символният низ, сочен от `word`, се съдържа като подниз в `text`. За определеност считаме, че празният низ се съдържа във всеки друг.

```
bool g(const char* text, const char* word)
{
  if (!*word) return true;
  if (!*text) return _______________;
  if (*word != *text) return false;
  return g(_______________, _______________);
}

bool f(const char* text, const char* word)
{
  if (!*word) return _______________;
  if (!*text) return false;
  return g(_______________, _______________) ||
         f(_______________, _______________);
}
```

5) Да се посочи какво ще изведе на екрана даденият по-долу фрагмент:

```
char arr[3][3] = { 'a', 'b', 'c',
                   'd', 'e', 'f',
                   'g', 'h', 'i' };
for (int i = 0; i < 3; ++i)
  std::cout << arr[2-i][i];
```

_____

6) Да се посочи какво ще изведе на екрана даденият по-долу фрагмент:

```
double var = 5 / 2;
std::cout << var;
```

_____

Критерии за оценяване

- Точки се дават само за напълно коректно посочени отговори.
- В подточките, в които се изисква да се посочи какво ще се изведе, точки се дават само ако отговорът напълно съвпада с това, което извежда съответният код. В противен случай се дават 0 т.
- В задачите за посочване на стойност на израз, ако отговорът не съвпада напълно с коректното решение се дават нула точки.
- В задачите за посочване на стойност на израз, ако вместо булевите литерали true/false се посочат числата 1/0, точките се намаляват наполовина.
- В задачата за довършване на кода на функцията, ако написаното не е синтактично или логически коректно или е различно от коректния отговор, се дават 0 т.
- Сумата от точките се закръгля до цяло число.

Максималната оценка за всяка подточка е както следва:

- Подточка 1: 2 точки (по 0.5 точки за всеки напълно коректен отговор)
- Подточка 2: 1 точка
- Подточка 3: 1 точка (0.5 точки, ако коректно е посочен `strcpy` за първата функция; 0.5 точки, ако коректно е посочена `strcat` за вторите две извиквания).
- Подточка 4: 4 точки (по 0.5 точки за всеки напълно коректен отговор).
- Подточка 5: 1 точка
- Подточка 6: 1 точка (отговорът се счита за коректен, независимо дали е посочен с десетична точка; например 2, 2.0 или 2,0 са коректни отговори).

Примерно решение:

1)

```
arr[1] == *(arr+2) --> false
arr == &arr[0] --> true
(arr+1) == &arr[1] --> true
*arr == arr[0] --> true
```

2)

```
ace
```

3)

```
strcpy(result, s1);
strcat(result, " ");
strcat(result, s2);
```

4)

```
bool g(const char* text, const char* word)
{
  if (!*word) return true;
  if (!*text) return false;
  if (*word != *text) return false;
  return g(text + 1, word + 1);
}

bool f(const char* text, const char* word)
{
  if (!*word) return true;
  if (!*text) return false;
  return g(text, word) ||
         f(text + 1, word);
}
```

5)

```
gec
```

6)

```
възможни отговори: 2, 2.0, 2,0 и т.н.
```

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

1) Освен конструктора по подразбиране (default constructor), кои други функции влизат в "голямата четворка" (функциите от т.нар. "rule-of-3")? Да се попълнят имената им в полетата долу:

_____
_____
_____

2) Нека е дадена дефиницията:

```
class foo {
public:
  virtual void f() {};
  void g() {};
};
```

Срещу всеки от редовете, които извикват `f` или `g`, да се запише "статично" или "динамично" според вида свързване, който ще се използва за тях.

```
foo obj;
foo& ref = obj;
obj.f();  ____________________
obj.g();  ____________________
ref.f();  ____________________
ref.g();  ____________________
```

3) Нека са дадени следните дефиниции:

```
class base {
public:  int a;
private: int b;
};
class derived : protected base { };
```

Да се посочи каква ще бъде видимостта на променливите `a` и `b` в класа `derived` – `public`, `protected` или `private`.

- Видимост на `a`: _____
- Видимост на `b`: _____

4) Нека класът `X` е абстрактен. Срещу всяко от твърденията да се посочи "да" или "не" според това дали е вярно:

- Могат да се създават обекти от тип `X`: ____
- Могат да се създават референции (reference) към обекти от тип `X`: ____

5) Нека е дадена следната дефиниция:

```
struct s {
public:
  static int var;
  s() { var = 5; }
};
int s::var = 0;
```

Да се посочи какво ще изведе следният фрагмент:

```
std::cout << '(' << s::var << ')';
s obj1;
obj1.var = 10;
s obj2;
std::cout << '-' << s::var << '-';
```

_____

6) Да се допълни дефиницията на класа `test`, така че функцията `f` да бъде чиста виртуална (pure-virtual) и класът да може коректно да се използва като основа на полиморфна йерархия.

```
class test {
public:
  _____________ void f() _______________;
  ____________________________________
};
```

7) Да се допълни дефиницията на шаблона `Array`, така че функцията `test` да се компилира коректно и да извежда на стандартния изход `55`.

```
_____________ <_____________>
class Array {
  static const size_t size = 10;
  T data[size];
public:
  _____________ at(size_t index) {
    if (index ______________________)
      throw std::out_of_range("error");
    return data[index];
  }
};

void test() {
  Array<int> a;
  a.at(0) = 5;
  std::cout << a.at(0);
  Array<Array<int>> b;
  b.at(0) = a;
  std::cout << b.at(0).at(0);
}
```

Критерии за оценяване

- Точки се дават само за напълно коректно посочени отговори.
- В подточките, в които се изисква да се посочи какво ще се изведе, точки се дават само ако отговорът напълно съвпада с това, което извежда съответният код. В противен случай се дават 0 т.
- В задачите за посочване на стойност на израз, ако отговорът не съвпада напълно с коректното решение, се дават нула точки.
- В задачите за посочване на стойност на израз, ако вместо булевите литерали `true` и `false` се посочат числата 1 и 0, точките се намаляват наполовина.
- В задачата за довършване на кода на функцията, ако написаното не е синтактично или логически коректно или е различно от коректния отговор, се дават 0 т.
- Сумата от точките се закръгля до цяло число.

Максималната оценка за всяка подточка е както следва:

1. 1 точка (1/3 точки за коректен отговор, −1/3 точки за грешно посочена функция, минимална оценка: 0 точки).
2. 2 точки (по 0.5 за коректен отговор).
3. 1 точка (по 0.5 за коректен отговор).
4. 1 точка (по 0.5 за коректен отговор).
5. 1 точка (по 0.5 съответно ако е посочено извеждането на 0 и на 5; Ако в отговора не са включени скобите и тиретата, оценката се намалява с 0.2 точки).
6. 2 точки (по 0.5 съответно за `virtual` и за `= 0` във функцията `f`; 1 точка за виртуален деструктор).
7. 2 точки (по 0.5 за всяко коректно попълнено място).

Примерно решение:

1)

- “копиращ конструктор” или “конструктор за копиране” или “copy constructor”
- “копиращо присвояване” или “операция/оператор за присвояване” или “copy assignment” или “assignment operator” или “`operator=`”
- “деструктор” или “destructor”

2)

```
foo obj;
foo& ref = obj;
obj.f();  статично
obj.g();  статично
ref.f();  динамично
ref.g();  статично
```

3)

- Видимост на `a`: `protected`
- Видимост на `b`: `private` или “няма видимост в производния клас”

4)

- Могат да се създават обекти от тип `X`: не
- Могат да се създават референции (reference) към обекти от тип `X`: да

5)

```
(0)-5-
```

6)

```
class test {
public:
  virtual void f() = 0;
  virtual ~test() {}
};
```

7)

```
template <typename T>
class Array {
  static const size_t size = 10;
  T data[size];
public:
  T& at(size_t index) {
    if (index >= size)
      throw std::out_of_range("error");
    return data[index];
  }
};
```

Задача 3. Разглежда се множество от системи линейни уравнения с реални коефициенти, където λ е реален параметър и свободните коефициенти b_1, b_2, b_3, b_4 са реални числа:

$$(*) \quad \left| \begin{array}{rcl} 4x_1 - 7x_2 + 6x_3 + 9x_4 &=& b_1 \\ -2x_1 + 11x_2 - 8x_3 + 3x_4 &=& b_2 \\ 3x_1 - 9x_2 + 7x_3 + 3x_4 &=& b_3 \\ x_1 - x_2 + x_3 + \lambda x_4 &=& b_4 \end{array} \right.$$

а) В случая на хомогенна система (т.е. когато $b_1 = b_2 = b_3 = b_4 = 0$) да се реши системата и да се определи фундаментална система от решения (ФСР) в зависимост от параметъра λ.

б) Нека B_3 е множеството, състоящо се от всички свободни стълбове, за които системата $(*)$ има решение, когато $\lambda = 3$, т.е.:

$$B_3 = \{(b_1, b_2, b_3, b_4) \mid \text{системата } (*) \text{ има решение при } \lambda = 3\}.$$

Да се докаже, че B_3 е линейно подпространство на 4–мерното пространство $\mathbb{R}^4$, да се определи размерността му и да се намери базис на B_3.

Примерно решение:

а) (5 т.) Преобразуваме матрицата на системата (без стълба от свободните членове, защото е нулев). Първо изваждаме третия ред от първия, както и прибавяме третия ред към втория ред.

$$A = \begin{pmatrix} 4 & -7 & 6 & 9 \\ -2 & 11 & -8 & 3 \\ 3 & -9 & 7 & 3 \\ 1 & -1 & 1 & \lambda \end{pmatrix} \sim \begin{pmatrix} 1 & 2 & -1 & 6 \\ 1 & 2 & -1 & 6 \\ 3 & -9 & 7 & 3 \\ 1 & -1 & 1 & \lambda \end{pmatrix}$$

Първия ред изваждаме от втория, първия ред вадим и от четвъртия и освен това първия ред умножен по 3 го вадим от третия. Третия ред го разделяме на 5, вадим от последния ред и подреждаме редовете. Получава се следната матрица.

$$A \sim \begin{pmatrix} 1 & 2 & -1 & 6 \\ 0 & 0 & 0 & 0 \\ 0 & -15 & 10 & -15 \\ 0 & -3 & 2 & \lambda - 6 \end{pmatrix} \sim \begin{pmatrix} 1 & 2 & -1 & 6 \\ 0 & -3 & 2 & -3 \\ 0 & 0 & 0 & \lambda - 3 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

В зависимост от стойността на параметъра λ имаме два случая:

Случай 1: **Когато** $\lambda \neq 3$**.** В този случай $\lambda - 3 \neq 0$ и матрицата има три линейно независими редове и нейния ранг е $r(A) = 3$. В този случай $x_4 = 0$, решението зависи от 1 реален параметър $x_3 = p$ и заместваики в първите два реда получаваме $x_2 = \frac{2p}{3}$ и $x_1 = \frac{-p}{3}$. Решението е едномерно линейно подпространство на четиримерното пространство $\mathbb{R}^4$ и общият вид на решението е

$$U_{\lambda \neq 3} = \left\{ \left(\frac{-p}{3}, \frac{2p}{3}, p, 0 \right) \mid p \in \mathbb{R} \right\}.$$

Фундаменталната система от решения е базис на подпространството от решения и се състои от един вектор и при $p = 3$ получаваме вектора $g = (-1, 2, 3, 0)$, който формира ФСР на решението в случая $\lambda \neq 3$.

Случай 2: **Когато** $\lambda = 3$**.** В този случай $\lambda - 3 = 0$ и матрицата има ранг 2. Решението зависи от два параметъра $x_3 = p$ и $x_4 = q$, заместваме и получаваме $x_2 = \frac{2p}{3} - q$ и $x_1 = -\frac{p}{3} - 4q$. Общият вид на решението, което е двумерно подпространство, е

$$U_{\lambda = 3} = \left\{ \left(-\frac{1}{3}p - 4q, \frac{2}{3}p - q, p, q \right)) \mid p, q \in \mathbb{R} \right\}.$$

Базис на решението, което е фундаментална система от решения, можем да получим когато дадем два набора независими стойности на параметрите, например при $p_1 = 0$ и $q_1 = 1$ получаваме $g_1 = (-4, -1, 0, 1)$ и при $p_2 = 3$ и $q_2 = 0$ получаваме $g_2 = (-1, 2, 3, 0)$. Векторите g_1, g_2 образуват фундаментална система от решения в случая $\lambda = 3$.

б) (5 т.) *Първи начин:* Нека стълбовете на матрицата на системата са c_1, c_2, c_3, c_4 и свободния стълб е b, където

$$c_1 = \begin{pmatrix} 4 \\ -2 \\ 3 \\ 1 \end{pmatrix}, c_2 = \begin{pmatrix} -7 \\ 11 \\ -9 \\ -1 \end{pmatrix}, c_3 = \begin{pmatrix} 6 \\ -8 \\ 7 \\ 1 \end{pmatrix}, c_4 = \begin{pmatrix} 9 \\ 3 \\ 3 \\ 3 \end{pmatrix}, b = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{pmatrix}.$$

От теоремата на Руше е известно, че системата $(*)$ има решение тогава и само тогава, когато рангът на матрицата на системата е равен на ранга на разширената матрица. За произволна

матрица знаем, че нейният ранг е равен на ранга на стълбовете на матрицата, който пък е равен на размерността на линейната обвивка на вектор-стълбовете:

$$b \in B_3 \Leftrightarrow r(A) = r(\overline{A}) \Leftrightarrow r(c_1, \dots c_4) = r(c_1, \dots, c_4, b) \Leftrightarrow \dim \ell(c_1, \dots, c_4) = \dim \ell(c_1, \dots, c_4, b).$$

Това е възможно точно когато стълбът от свободните членове принадлежи на линейната обвивка на стълбовете на системата:

$$b \in B_3 \Leftrightarrow b \in \ell(c_1, \dots, c_4) \Longrightarrow B_3 = \ell(c_1, \dots, c_4).$$

Оттук получаваме, че множеството B_3 е линейно подпространство и от пресмятанията в т. а) получаваме

$$\dim B_3 = r(c_1, \dots, c_4) = r(A) = 2.$$

Базис на това двумерно подпространство ще намерим като вземем произволни два линейно независими вектора измежду стълбовете $c_1, \dots, c_4$. Например един базис е

$$B_3 = \ell(c_1, c_2), \quad \text{където} \quad c_1 = \begin{pmatrix} 4 \\ -2 \\ 3 \\ 1 \end{pmatrix}, c_2 = \begin{pmatrix} -7 \\ 11 \\ -9 \\ -1 \end{pmatrix}.$$

Втори начин: Преобразуваме матрицата заедно със стълба от свободните членове. Използвайки четвъртия ред анулираме всички елементи на първо място от останалите редове. С новополучения първия ред анулираме лявата част на втори и трети ред:

$$\left(\begin{array}{cccc|c} 4 & -7 & 6 & 9 & b_1 \\ -2 & 11 & -8 & 3 & b_2 \\ 3 & -9 & 7 & 3 & b_3 \\ 1 & -1 & 1 & 3 & b_4 \end{array}\right) \sim \left(\begin{array}{cccc|c} 0 & -3 & 2 & -3 & b_1 - 4b_4 \\ 0 & 9 & -6 & 9 & b_2 + 2b_4 \\ 0 & -6 & 4 & -6 & b_3 - 3b_4 \\ 1 & -1 & 1 & 3 & b_4 \end{array}\right) \sim \left(\begin{array}{cccc|c} 0 & -3 & 2 & -3 & b_1 - 4b_4 \\ 0 & 0 & 0 & 0 & 3b_1 + b_2 - 10b_4 \\ 0 & 0 & 0 & 0 & -2b_1 + b_3 + 5b_4 \\ 1 & -1 & 1 & 3 & b_4 \end{array}\right)$$

Системата има решение точно когато дясната част на втори и трети ред е равна на нула

$$b \in B_3 \Leftrightarrow \left| \begin{array}{ll} 3b_1 + b_2 - 10b_4 & = 0 \\ -2b_1 + b_3 + 5b_4 & = 0 \end{array} \right.$$

Това е хомогенна система и множеството B_3 е нейно решение, което е подпространство на $\mathbb{R}^4$. Даваме два независими набора от стойности за b_1, b_4 и определяме стойностите на b_2, b_3 и по този начин намираме базис на подпространството B_3:

$$\begin{array}{llll} \text{ако} & b_1 = 1, & b_4 = 0 & \Rightarrow \quad b_2 = -3, \quad b_3 = 2 \\ \text{ако} & b_1 = 0, & b_4 = 1 & \Rightarrow \quad b_2 = 10, \quad b_3 = -5 \end{array}$$

Получихме, че един базис на пространството B_3 образуват векторите $t_1 = (1, -3, 2, 0)$ и $t_2 = (0, 10, -5, 1)$.

Задача 4. Крайно кореново дърво, всеки връх на което съдържа цяло число и може да има произволен брой деца, се представя в Scheme като наредена двойка, състояща се от стойността на корена и списък от директни поддървета, а в Haskell — със следната рекурсивна структура:

```
data Tree = T { root :: Int, subtrees :: [Tree] } deriving Show
```

Листо наричаме дърво с единствен връх, **клонка** наричаме дърво, чиито директни поддървета са листа, а **пръчка** наричаме дърво, в което всеки връх има най-много едно дете. Казваме, че едно дърво се **подрязва**, ако от него се премахнат всички клонки, с изключение на корена, и че се **окастря**, ако всички пръчки в него, които са с повече от два върха и не са част от други пръчки, се скъсят откъм листата до дължина точно два върха. Да се попълнят празните полета по-долу, така че:

- функциите `leaf`, `twig` и `stick` да проверяват дали дърво е съответно листо, клонка или пръчка;
- функциите `trim` и `prune` да връщат съответно подрязано или окастрено копие на дървото, подадено им като параметър.

Упътване: *могат да се използват наготово* `all`, `any`, `car`, `cdr`, `filter`, `foldl`, `foldr`, `head`, `null`, `null?`, `tail`, `length`, `map`, *както и всички функции в R5RS за Scheme и в Prelude за Haskell.*

Haskell

```
leaf  _____
twig  _____
stick _____
_____

trim (T x ts) = T x _____
                    _____

prune t@(T x []) = _____
prune t@(T x ts) = T x (if stick t
                        then _____
                        else _____
```

Пример:

```
twig (T 1 [T 2 [], T 3 []]) ⟶ True          stick (T 1 [T 2 [T 3 [T 4 []]]]) ⟶ True
tree = T 1 [T 2 [T 3 []], T 4 [T 5 [T 6 []]], T 7 [T 8 [], T 9 [T 10 [T 11 []]]]]
trim tree  ⟶ T 1 [T 4 [], T 7 [T 9 []]]
prune tree ⟶ T 1 [T 2 [T 3 []], T 4 [T 5 []], T 7 [T 8 [], T 9 [T 10 []]]]
```

Scheme

```
(define (leaf t)  _____)
(define (twig t)  _____)
(define (stick t) _____)
(define (trim t)
  (cons (car t) _____
                _____))
(define (prune t)
  (if (leaf t)  _____
      (cons (car t) (if (stick t)
                        _____
                        _____))))
```

Пример:

```
(twig '(1 (2) (3))) ⟶ #t                    (stick '(1 (2 (3 (4))))) ⟶ #t
(define tree '(1 (2 (3)) (4 (5 (6))) (7 (8) (9 (10 (11))))))
(trim tree)  ⟶ (1 (4) (7 (9)))
(prune tree) ⟶ (1 (2 (3)) (4 (5)) (7 (8) (9 (10))))
```

Примерни решения

Haskell

```
leaf = null . subtrees

twig = all leaf . subtrees

stick (T _ [t]) = stick t
stick (T _ ts)  = null ts

trim (T x ts) = T x [ trim t | t <- ts, not $ twig t ]

prune t@(T x []) = t
prune t@(T x ts) = T x (if stick t
                        then let [T y _] = ts in [T y []]
                        else map prune ts)
```

Scheme

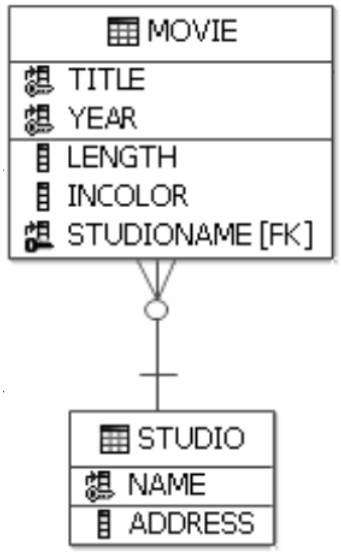
```
(define (leaf t)
  (null? (cdr t)))

(define (twig t)
  (null? (filter (lambda (x) (not (leaf x))) (cdr t))))

(define (trim t)
  (cons (car t) (map trim (filter (lambda (x) (not (twig x))) (cdr t)))))

(define (stick t)
  (or (leaf t) (and (null? (cddr t)) (stick (cadr t)))))

(define (prune t)
  (if (leaf t) t
      (cons (car t) (if (stick t)
                        (list (list (caadr t)))
                        (map prune (cdr t))))))
```

Критерии за оценяване

Ако е писано и по двата езика, взима се по-високият резултат. Сумата от точките се закръгля нагоре до цяло число.

- **0,5 т.** за коректна реализация на `leaf`;
  - **0 т.** за реализация на Haskell, която използва непълно съпоставяне с образци (pattern matching) само с `T _ []`, без да разглежда случая за непразен списък от директни поддървета;
- **1,5 т.** за коректна реализация на `twig`;
  - **0,5 т.** ако се прави проверка само за първото едно или две директни поддървета;
- **1,5 т.** за коректна реализация на `stick`;
- **3,0 т.** за коректна реализация на `trim`, от които:
  - **1,0 т.** за коректно извикване на `filter`;
  - **1,0 т.** за коректно извикване на `map`;
  - **1,0 т.** за коректна последователност на `map` след `filter`;
- по ред на празните слотове на `prune`:
  1. **0,5 т.** за `t` или `T x []` за Haskell;
  2. **1,5 т.** за коректно конструиране на листо със стойността на единственото дете на `t`;
  3. **1,5 т.** за коректно рекурсивно извикване на `prune` за всички директни поддървета на `t`.

Задача 5. Дадена е базата от данни `Movies`, в която се съхранява информация за филми и филмови студиа, които ги произвеждат.



Таблицата `Studio` съдържа информация за филмови студиа:

- `name` — име, първичен ключ
- `address` — адрес;

Таблицата `Movie` съдържа информация за филми. Атрибутите `title` и `year` заедно формират първичния ключ.

- `title` — заглавие
- `year` — година, в която е заснет филмът
- `length` — дължина в минути
- `incolor` — `'Y'` за цветен филм и `'N'` за черно-бял
- `studioname` — име на студио, външен ключ към `Studio.name`;

1) Да се напише заявка, която извежда заглавията и дължините в минути на всички цветни филми без най-дългия цветен и без тези с неизвестна дължина. Ако има няколко филма с максимална дължина, нито един от тях не трябва да бъде изведен.

2) Да се посочи коя от следните заявки извежда имената на тези студиа, които нямат филми или са снимали филми само в една единствена година:

А)
```
SELECT name
FROM Studio
WHERE name NOT IN (SELECT studioname
                   FROM Movie)
      OR COUNT(DISTINCT year) = 1;
```

Б)
```
SELECT studioname
FROM (SELECT studioname, year
      FROM Movie
      GROUP BY studioname, year) Years
GROUP BY studioname
HAVING COUNT(*) <= 1;
```

В)
```
SELECT name
FROM Studio
LEFT JOIN Movie ON name = studioname
GROUP BY name
HAVING COUNT(DISTINCT year) <= 1;
```

Г)
```
SELECT name
FROM Studio
WHERE NOT EXISTS (SELECT * FROM Movie)
UNION
SELECT studioname
FROM Movie
GROUP BY studioname
HAVING COUNT(DISTINCT year) = 1;
```

Примерно решение на подзадача 1:

```
SELECT title, length
FROM Movie
WHERE incolor = 'Y'
      AND length < (SELECT MAX(length)
                    FROM Movie
                    WHERE incolor = 'Y');
```

Критерии за оценяване:

1) Общо 5 т., от които
   - 1 т. за коректни `SELECT` и `FROM` клаузи на външната заявка, както и `WHERE` клауза на външната заявка без проверка на дължина;
   - 3 т. за коректна подзаявка за намиране на най-дългите цветни филми;
     - ако подзаявката намира най-дългите филми изобщо, се дава само 1 т. за този критерий;
     - ако подзаявката намира точно един най-дълъг филм, се дават само 2 т. за този критерий;
   - 1 т. за правилна релация с резултата от подзаявката (в конкретния пример това са `<`, `<>`, `!=`, `NOT IN`).

2) Единственият верен отговор е B). Посочването на този отговор носи 5 т., а посочването на грешен отговор или комбинация от отговори носи 0 т.

Задача 6. Информационна система съхранява информация за доставки на пратки в куриерска фирма Споки. Съхранява се информация за клиентите на куриерската фирма: телефонен номер на клиент – низ точно 10 символа, уникален за всеки клиент, име на клиент – низ до 50 символа. Съхранява се информация и за офисите на куриерската фирма: име на офиса – низ до 30 символа, уникално за всеки офис, град в който се намира офисът – низ до 20 символа и улица – низ до 30 символа.

Клиентите на куриерската фирма могат да изпращат и получават пратки. За пратките се пази информация за номер на пратка – низ точно 10 символа, уникален за всяка пратка, адрес, на който пратката трябва да бъде доставена – низ до 30 символа, статус на пратката – низ точно 10 символа и дължима такса за доставка на пратката – реално число. Един клиент може да изпраща много пратки, но една пратка може да бъде изпратена само от един клиент. В системата трябва да се пази информация за дата на изпращане на пратката. Един клиент може да получава много пратки, но една пратка може да бъде получена само от един клиент. В системата трябва да се пази информация за дата на получаване на пратката. Физически пратката се оставя в офис на куриера – офис на подателя, и пристига в офис на куриер – офис на получателя. В един офис може да има много пратки.

В системата се пази информация и за куриерите доставчици на пратки. За всеки доставчик се пази номер на куриер – низ точно 5 символа, уникален за всеки доставчик и име на доставчик – низ до 30 символа. Един доставчик може да разнася много пратки, но една пратка се доставя до адрес от един доставчик. Доставчиците работят в точно един офис, а в един офис може да работят много доставчици.

Задание:

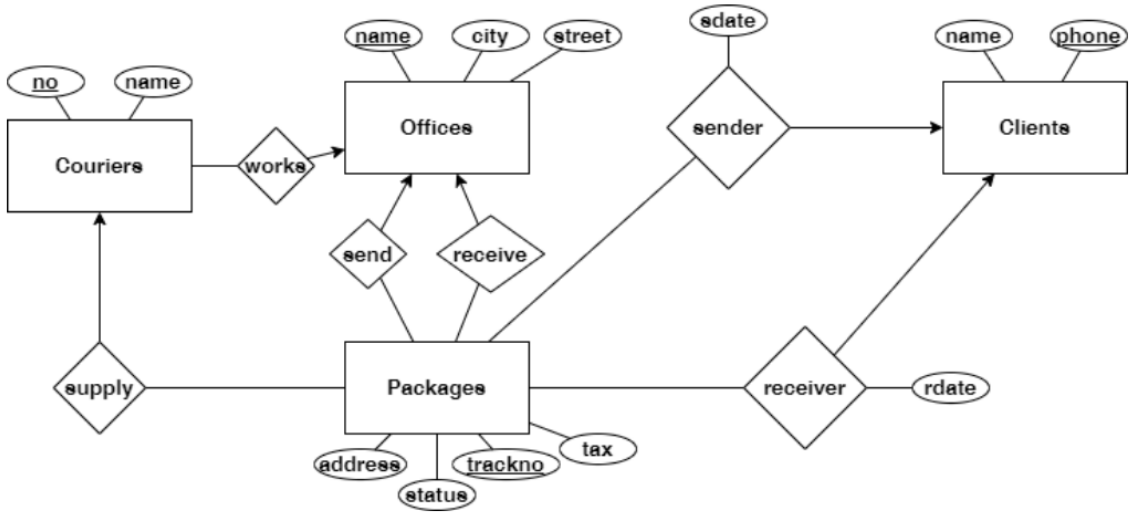
а) Да се направи E/R модел на БД, която съхранява гореописаната информация. Да се начертае E/R диаграма на модела.

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.

Примерно решение:

а)



б) Offices (name, city, street)
Couriers (no, name, officeName)
Clients (phone, name)
Packages (trackno, address, tax, status, officeSend, officeReceive, courierNo, clientSender, dateSend, clientReceiver, dateRecieve)

в)
```
CREATE TABLE Offices (
    name VARCHAR(30) NOT NULL PRIMARY KEY,
    city VARCHAR(20) NOT NULL,
    street VARCHAR(30) NOT NULL);
CREATE TABLE Couriers (
    no CHAR(5) NOT NULL PRIMARY KEY,
    name VARCHAR(30) NOT NULL,
    officeName VARCHAR(30) REFERENCES Offices(NAME)
);
CREATE TABLE Clients (
    phone CHAR(10) NOT NULL PRIMARY KEY,
    name VARCHAR(50));
CREATE TABLE Packages (
    trackno CHAR(10) NOT NULL PRIMARY KEY,
    address VARCHAR(30) NOT NULL,
    tax DECIMAL(9,2),
    status CHAR(10),
    officeSend VARCHAR(30) REFERENCES Offices(NAME),
    officeReceive VARCHAR(30) REFERENCES Offices(NAME),
    courierNo CHAR(5) REFERENCES Couriers(NO),
    clientSender CHAR(10) REFERENCES CLIENTS(PHONE),
    dateSend DATE,
    clientReceiver CHAR(10) REFERENCES CLIENTS(PHONE) ,
    dateRecieve DATE
);
```

Критерии за оценяване:

а) 4 т. от които:

- до 2 т. за коректно отразяване на информацията в условието на задачата, като
  - 1 т. се дават за частично отразяване на информацията
  - 0 т. се дават за основни грешки в отразяване на информацията
- до 2 т. за коректно реализирана диаграма, като
  - 1 т. се дават за пропуски в E/R диаграмата
  - 0 т. се дават за съществени пропуски в E/R диаграмата

б) 3 т. от които:

- 1 т. за коректно преобразуване на множествата от същности към релации
- +2 т. за коректно представени и оптимизирани връзки “едно към много”

в) 3 т. от които:

- 1 т. за коректно описание на колоните и техния тип
- +2 т. за коректно описание на първичните и външните ключове

Задача 7. Разглеждаме Информационната система за пратки, представена в Задача 6. Клиент може да заяви пратка чрез системата и по-късно да предаде пратката на служител в офис или на заявен до собствения адрес куриер. Ако не използва системата, клиентът посещава офис или заявява по телефона куриер, за да се извършат всички формалности и да предаде пратката. При заявяване на пратката чрез системата, клиентът попълва данни за подателя и получателя, за пратката и свързани с нея услуги (напр. обявена стойност, наложен платеж) и се изчислява цената. Клиентът има 3 дни, за да предаде пратката в офис или на куриер. Куриерът предава събраните пратки в офиса на подателя. От офиса на подателя пратката се транспортира до офиса на получателя. След като достигне до офиса на получателя, до получателя се изпраща уведомление за пристигнала пратка. Получателят разполага със 7 дни от датата на уведомяване, за да приеме пратката в офиса на получателя или от куриер. Ако не я приеме в този срок, пратката се изпраща в офиса на подателя, за да му бъде върната.

а) Да се опише в пълен формат потребителски случай за заявяване на пратка от клиент. В описанието да се включат: актьори; предусловия; следусловия (резултати); основен сценарий; отношения с други потребителски случаи (ако е приложимо); алтернативен сценарий.

б) Да се подготви диаграма на състоянията на обект “Пратка”. Диаграмата да се подготви в термините на UML стандарта.

Примерно решение:

а)

Актьори	Клиент
Предусловия	UC Вход в системата
Следусловия	Обектът “Пратка” е в състояние “Заявена от подател”
Основен сценарий	
1	Клиентът избира опция “Нова пратка”
2	Система отваря форма за пратка
3	Клиентът попълва форма за пратка
4	Клиентът избира опция “Изчисляване на цена”
5	includes UC “Изчисляване на цена”
6	Системата извежда за преглед цялата информация за пратка
7	Клиентът избира опция “Изпрати”
8	Системата извежда съобщение за успешно създадена пратка
9	Обектът “Пратка” преминава в състояние “Заявена от подател”
Алтернативен сценарий	
3.1	Ако Избран офис = “врата на клиента”, extended by UC “Заявяване на куриер” Системата извежда съобщение за успешно заявен куриер
3.2	Изпълнението се връща в т.3 от основния сценарий

б)

Критерии за оценяване:

а) точките се изчисляват по следните критерии и се закръглят до цяло число:

- съответствие с описания сценарий: 0,30 т.
- съответствие с условието на задачата: 0,30 т.
- съответствие с изискванията за подготовка на модел на потребителските случаи (ПС) в частта пълно описание на ПС, включително:
  - коректно са определени актьори: 0,30 т.
  - коректно са определени предусловия и следусловия: 0,30 т.
  - коректно е определена функционалността на системата, която обхваща отделният потребителски случай: 2,50 т.
  - основен и алтернативен сценарии са описани в отделни стъпки и всеки от тях представлява напълно завършен сценарий: 0,30 т.
  - отношенията с други ПС са посочени в предусловие или на стъпките от сценария, където се извършва разклонение или извикване: 1,00 т.

б) точките се изчисляват по следните критерии и се закръглят до цяло число:

- съответствие с описания сценарий: 3,50 т.
- съответствие с условието на задачата: 0,50 т.
- съответствие с UML нотацията: 1,00 т.

Задача 8. Да се намери неопределеният интеграл

$$\int \left(x + \operatorname{tg}^2 x\right) \sin^2 x \, dx, \quad x \in \left(-\frac{\pi}{2}, \frac{\pi}{2}\right).$$

Примерно решение:

Започваме със свеждането

$$\int \left(x + \operatorname{tg}^2 x\right) \sin^2 x \, dx = \int x \sin^2 x \, dx + \int \frac{\sin^4 x}{\cos^2 x} \, dx. \qquad (*)$$

Намираме всеки от двата неопределени интеграла горе вдясно. За първия имаме

$$\begin{aligned}
\int x \sin^2 x \, dx &= \frac{1}{2} \int x(1 - \cos 2x) \, dx = \frac{1}{2} \int x \, dx - \frac{1}{4} \int x \cos 2x \, d(2x) \\
&= \frac{x^2}{4} + \text{const} - \frac{1}{4} \int x \, d(\sin 2x) \quad \text{(внасяме } \cos 2x \text{ под знака на диференциала)} \\
&= \frac{x^2}{4} + \text{const} - \frac{1}{4} \left(x \sin 2x - \int \sin 2x \, dx \right) \quad \text{(интегрираме по части)} \\
&= \frac{x^2}{4} - \frac{1}{4} x \sin 2x - \frac{1}{8} \cos 2x + \text{const}.
\end{aligned}$$

За втория интеграл вдясно на $(*)$ имаме

$$\begin{aligned}
\int \frac{\sin^4 x}{\cos^2 x} \, dx &= \int \frac{\left(1 - \cos^2 x\right)^2}{\cos^2 x} \, dx \\
&= \int \frac{dx}{\cos^2 x} - 2 \int dx + \int \cos^2 x \, dx \\
&= \operatorname{tg} x - 2x + \text{const} + \frac{1}{2} \int (1 + \cos 2x) \, dx \\
&= \operatorname{tg} x - 2x + \text{const} + \frac{1}{2} \int dx + \frac{1}{4} \int \cos 2x \, d(2x) \\
&= \operatorname{tg} x - 2x + \frac{x}{2} + \frac{1}{4} \sin 2x + \text{const} \\
&= \operatorname{tg} x - \frac{3x}{2} + \frac{1}{4} \sin 2x + \text{const}.
\end{aligned}$$

Окончателно получаваме

$$\int \left(x + \operatorname{tg}^2 x\right) \sin^2 x \, dx = \frac{x^2}{4} - \frac{3x}{2} - \frac{1}{4} x \sin 2x + \frac{1}{4} \sin 2x - \frac{1}{8} \cos 2x + \operatorname{tg} x + \text{const}.$$

Критерии за оценяване: Общо 10 т., от които:

- за намиране на $\int x \sin^2 x \, dx$: 4 т., в това число:
  - за понижаване на степента на sin: 1 т.,
  - за намиране на $\int x \, dx$: 1 т.,
  - интегриране по части: 2 т.;
- за намиране на $\int \frac{\sin^4 x}{\cos^2 x} \, dx$: 4 т., в това число:
  - подходящо разлагане на подинтегралната функция: 1 т.,
  - за намиране на $\int \frac{dx}{\cos^2 x}$: 1 т.,
  - за намиране на $\int dx$: 1 т.,
  - за намиране на $\int \cos^2 x \, dx$: 1 т.;
- окончателен отговор: 2 т. (при отсъствие на интеграционната константа **не** се присъждат).

Чернова