СОФИЙСКИ УНИВЕРСИТЕТ
“СВ. КЛИМЕНТ ОХРИДСКИ”

ФАКУЛТЕТ ПО МАТЕМАТИКА
И ИНФОРМАТИКА

ДЪРЖАВЕН ИЗПИТ

ЗА ПОЛУЧАВАНЕ НА ОКС “БАКАЛАВЪР ПО СОФТУЕРНО ИНЖЕНЕРСТВО”

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) Да се довърши кода на power така, че тя да пресмята повдигането на base на степен power.

```
int power(int base, unsigned int power)
{
  int result = ________________________;

  while(power > ___________________) {
    if(power % 2 == ___________) {
      _____________ = result * base;
    }

    ___________ /= 2;
    ___________ *= base;
  }

  return result;
}
```

2) Да се посочи какво ще изведе на екрана даденият по-долу фрагмент.

```
int a=10, b=010, c=0x10;
cout << a << ", " << b << ", " << c;
```

_____

3) Под всеки от фрагментите да се посочи какво ще изведе той на екрана.

```
int var = 0;
while(var++ < 5)
  cout << var;
```

_____

```
int var = 0;
do {
  cout << var;
} while(++var < 5);
```

_____

4) Да се посочи какво ще изведе фрагментът. Допускаме, че sizeof(unsigned int) == 4.

```
unsigned int x = 0x11335577;
unsigned char* ptr = (unsigned char*)&x;
++*ptr;
++*(ptr+1);
++ptr[2];
++3[ptr];
cout << hex << "0x" << x;
```

_____

5) Нека a и b са променливи от тип char. Да се посочи какъв е проблемът в дадения по-долу фрагмент.

```
char result = a + b;
```

_____

Да се посочи възможен начин, чрез който може да се реши този проблем.

_____

6) Нека са дадени следните дефиниции:

```
const size_t size = 3;
char box[size][size] = {
  'a', 'b', 'c',
  'd', 'e', 'f',
  'g', 'h', 'i'
};
```

Да се довършат фрагментите по-долу така, че да извеждат съответните последователности от букви.

```
// Трябва да извежда aei
for (int i = 0; i < size; i++)
  cout << box[_______________][_______________];

// Трябва да извежда ceg
for (int i = 0; i < size; i++)
  cout << box[_______________][size__________];

// Трябва да извежда gec
for (int i = 0; i < size; i++)
  cout << box[size__________][_______________];

// Трябва да извежда abcfihgd
for(int i = 0; i < size; ++i)
  cout << box[0][i];
for(int i = __________; i < size; __________)
  cout << box[i][size-1];
for(int i = size_______; i >= 0; --i)
  cout << box[size-1][i];
for(int i = size-2; i _____ 0; --i)
  cout << box[i][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 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. 3 точки (по 0,5 за всяко коректно попълнено място);
2. 1 точка (по 0,3 за всяка коректно посочена стойност и още 0,1 ако са посочени и запетаите);
3. 1 точки (по 0,5 за всеки фрагмент);
4. 1 точка;
5. 1 точка (по 0,5 за всеки подвъпрос);
6. 3 точки (по 0,3 за всяко коректно попълнено място);

Примерно решение:

1)

```
int power(int base, unsigned int power)
{
  int result = 1;

  while(power > 0) {
    if(power % 2 == 1) {
      result = result * base;
    }

    power /= 2;
    base *= base;
  }

  return result;
}
```

2) `10, 8, 16`

3) `12345` за първия фрагмент и `01234` за втория фрагмент.

4) `0x12345678`

5) Възможно е резултатът от събирането на двете променливи да не се побира в променлива от тип `char`. Например, ако допуснем `sizeof(char) == 1` и също, че двете променливи съдържат стойността 100.

Проблемът може да се реши като за резултата се използва променлива от тип, който може да побере всевъзможните стойности, които биха се получили при събирането.

6)

```
// Трябва да извежда aei
for (int i = 0; i < size; i++)
  cout << box[i][i];

// Трябва да извежда ceg
for (int i = 0; i < size; i++)
  cout << box[i][size-i-1];

// Трябва да извежда gec
for (int i = 0; i < size; i++)
  cout << box[size-i-1][i];

// Трябва да извежда abcfihgd
for(int i = 0; i < size; ++i)
  cout << box[0][i];
for(int i = 1; i < size; ++i)
  cout << box[i][size-1];
for(int i = size-2; i >= 0; --i)
  cout << box[size-1][i];
for(int i = size-2; i > 0; --i)
  cout << box[i][0];
```

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

Документ наричаме символен низ с произволна дължина, който има име, също символен низ. Текстът в документа е съставен от редове, разделени със символа `'\n'`.

Папка наричаме контейнер, който може да съдържа както документи, така и други папки. Казваме, че `subfolder` е **вложена папка** във `folder`, ако `subfolder` е елемент на контейнера `folder` или е вложена в някоя папка, която е вложена във `folder`.

В следната програмата на езика за програмиране C++ липсват части.

Класът `Document` описва документ, а класът `Folder` описва папка. Класът `File` е абстрактен базов клас за `Document` и `Folder`, който дефинира операцията `search` за търсене на символен низ в йерархия от документи и папки.

Резултат от търсенето на низа `str` в даден документ `doc` наричаме такава тройка `(name,N,line)`, където `name` е името на документа `doc`, `line` е съдържанието на някой ред в документа, съдържащ `str` като подниз, а `N` е поредния номер на `line` в документа.

Резултатът от `doc.search(str)` е вектор с всички резултати от търсенето на `str` в `doc`.

При търсене в папка `folder`, `folder.search(str)` е вектор с всички резултати от търсенето на `str` в документите, съдържащи се във `folder` или в нейните вложени папки.

Да се попълнят липсващите части в програмата. При правилно заместване на празните места, програмата ще изведе следното на стандартния изход:

```
employees.txt, line 1: John Smith
employees.txt, line 2: Jane Smith
```

Да се приеме, че класът `Folder` не е нужно да прави копие на вложените обекти. Можете да използвате всякакви библиотечни функции като допишете съответните `#include` директиви.

```
#include <vector>
#include <iostream>


struct SearchResult
{
    std::string fileName;
    unsigned line_number;
    std::string line;
};
class File
{   public:

    ____________ std::vector<SearchResult> search(____________) const ____________

    virtual ~File(){}
};
class Document: public File
{   std::string name;
    std::string contents;
    public:
    Document(____________ _name,____________ _contents):_____________________________

    std::vector<SearchResult> search(____________ str) const
    {
        std::vector<SearchResult> result;
```

```
        return result;
    }
};
class Folder: public File
{
    ____________________  files;
    std::string name;
    public:
    Folder(____________ _name):____________________

    void addFile(____________ f)
    {files.push_back(f);}

    std::vector<SearchResult> search(____________ str) const
    {
        std::vector<SearchResult> result;









        return result;
    }
};
int main()
{   Document d1("employees.txt","John Smith\nMaryia Ivanova\n"),
             d2("inventory.txt","Computers: 3\nPrinters: 1"),
             d3("employees.txt","Ivan Petrov\nJane Smith"),
             d4("inventory.txt","Computers: 5, 3D Printers: 1");
    Folder root("root"), acme("ACME Soft, Inc."), best("Best Soft, OOD");
    acme.addFile(&d1); acme.addFile(&d2);
    best.addFile(&d3); best.addFile(&d4);
    root.addFile(&acme); root.addFile(&best);
    std::vector<SearchResult> results = root.search("Smith");
    for(unsigned i = 0; i < results.size(); ++i)
    {
        std::cout << results[i].fileName
                  << ", line " << results[i].line_number << ": "
                  << results[i].line << std::endl;
    }
}
```

Примерно решение

```
#include <vector>
#include <iostream>
#include <sstream>
#include <iostream>
struct SearchResult
{
    std::string fileName;
    unsigned line_number;
    std::string line;
};
class File
{
    public:
    virtual std::vector<SearchResult> search(const std::string&) const=0;
};
class Document: public File
{
    std::string name;
    std::string contents;
    public:
    Document(const std::string& _name,const std::string& _contents)
              :name(_name),contents(_contents){}
    std::vector<SearchResult> search(const std::string& s) const
    {
        std::istringstream doc(contents);
        std::string line;
        unsigned lineCount=0;
        std::vector<SearchResult> result;
        while(std::getline(doc,line))
        {
            ++lineCount;
            if (line.find(s) != std::string::npos)
            {
                result.push_back({name,lineCount,line});
            }
        }
        return result;
    }
};
class Folder: public File
{
    std::vector<File*> files;
    std::string name;
    public:
    Folder(const std::string &_name):name(_name){}
    void addFile(File *f)
    {files.push_back(f);}
    std::vector<SearchResult> search(const std::string& s) const
    {
        std::vector<SearchResult> result;
        for(const File * f:files)
```

```
        {
            std::vector<SearchResult> found = f->search(s);
            result.insert(result.end(),found.begin(),found.end());
        }
        return result;
    }
};
int main()
{
    Document d1("employees.txt","John Smith\nMaryia Ivanova\n"),
             d2("inventory.txt","Computers: 3\nPrinters: 1"),
             d3("employees.txt","Ivan Petrov\nJane Smith"),
             d4("inventory.txt","Computers: 5, 3D Printers: 1");
    Folder root("root"), acme("ACME Soft, Inc."), best("Best Soft, OOD");
    acme.addFile(&d1); acme.addFile(&d2);
    best.addFile(&d3); best.addFile(&d4);
    root.addFile(&acme); root.addFile(&best);
    std::vector<SearchResult> results = root.search("Smith");
    for(unsigned i = 0; i < results.size(); ++i)
    {
        std::cout << results[i].fileName
                  << ", line " << results[i].line_number << ": "
                  << results[i].line << std::endl;
    }
}
```

Критерии за оценяване

Задачата се оценява по долната схема, като сумата на точките се дели на 3 и се закръгля до цяло число.

```
#include <vector>
#include <iostream>
struct SearchResult
{
    std::string fileName;
    unsigned line_number;
    std::string line;
};
class File
{
    public:

    ____________ std::vector<SearchResult> search(____________) const ____________
    [*] virtual ...(std::string&) = 0;       +2 т.
    [*] друго вярно                          +1 т.
};
class Document: public File
{
    std::string name;
    std::string contents;


    public:
```

```
    Document(_____ _name,_____ _contents):_____
    [*] const std::string&   ИЛИ              +2 т.
    [*] друго вярно                           +1 т.

    [*] правилна инициализация                +2 т.

    std::vector<SearchResult> search(_____ str) const
    {
        std::vector<SearchResult> result;









    [*] вярна реализация                      +8 т.
        return result;
    }
};
class Folder: public File
{
    _____  files;
    [*] std::vector<File*>                    +2 т.

    std::string name;
    public:
    Folder(_____ _name):_____
    [*] const std::string&   ИЛИ              +2 т.
    [*] друго вярно                           +1 т.

    [*] правилна инициализация                +2 т.

    void addFile(_____ f)
    [*] File*                                 +2 т.
    {files.push_back(f);}

    std::vector<SearchResult> search(_____ str) const
    {
        std::vector<SearchResult> result;
```

```
    [*] вярна реализация                    +8 т.
        return result;
    }
};
int main()
{
    Document d1("employees.txt","John Smith\nMaryia Ivanova\n"),
             d2("inventory.txt","Computers: 3\nPrinters: 1"),
             d3("employees.txt","Ivan Petrov\nJane Smith"),
             d4("inventory.txt","Computers: 5, 3D Printers: 1");
    Folder root("root"), acme("ACME Soft, Inc."), best("Best Soft, OOD");
    acme.addFile(&d1); acme.addFile(&d2);
    best.addFile(&d3); best.addFile(&d4);
    root.addFile(&acme); root.addFile(&best);
    std::vector<SearchResult> results = root.search("Smith");
    for(unsigned i = 0; i < results.size(); ++i)
    {
        std::cout << results[i].fileName
                  << ", line " << results[i].line_number << ": "
                  << results[i].line << std::endl;
    }
}
```

Задача 3.

а) Да се намери рангът на матрицата

$$A = \begin{pmatrix} -1 & 5 & -2 & 7-\lambda & -1 \\ 1 & -5 & 17 & 0 & 0 \\ 0 & -1 & 10 & 1 & 0 \\ 0 & -2 & 5 & 0 & 1 \\ 0 & -3 & 15 & \lambda-4 & 1 \end{pmatrix} \in M_5(\mathbb{R}),$$

в зависимост от стойностите на реалния параметър λ.

б) Да се намери хомогенна система линейни уравнения, такава че пространството W от решенията ѝ, да съвпада с линейната обвивка на вектор-редовете на матрицата A, ако $\lambda = 5$.

в) Да се намери за кои стойности на реалните параметри α и β векторът $\mathbf{a} = (1, \alpha, 12, \beta, 1)$ принадлежи на подпространството $W \leq \mathbb{R}^5$ от условие б).

Примерно решение

а) Извършваме елементарни Гаусови преобразувания върху матрицата A, които я преобразуват последователно до необходимия еквивалентен вид, без да променят търсения ранг.

$$A = \begin{pmatrix}\mathbf{a}_1\\ \mathbf{a}_2\\ \mathbf{a}_3\\ \mathbf{a}_4\\ \mathbf{a}_5\end{pmatrix} = \begin{pmatrix}-1 & 5 & -2 & 7-\lambda & -1\\ 1 & -5 & 17 & 0 & 0\\ 0 & -1 & 10 & 1 & 0\\ 0 & -2 & 5 & 0 & 1\\ 0 & -3 & 15 & \lambda-4 & 1\end{pmatrix} \sim \begin{pmatrix}1 & -5 & 17 & 0 & 0\\ 0 & -1 & 10 & 1 & 0\\ 0 & -2 & 5 & 0 & 1\\ 0 & -3 & 15 & \lambda-4 & 1\\ -1 & 5 & -2 & 7-\lambda & -1\end{pmatrix}$$
$$\sim \begin{pmatrix}1 & -5 & 17 & 0 & 0\\ 0 & 1 & -10 & -1 & 0\\ 0 & 0 & -15 & -2 & 1\\ 0 & 0 & -15 & \lambda-7 & 1\\ 0 & 0 & 15 & 7-\lambda & -1\end{pmatrix} \sim \begin{pmatrix}1 & -5 & 17 & 0 & 0\\ 0 & 1 & -10 & -1 & 0\\ 0 & 0 & -15 & -2 & 1\\ 0 & 0 & 0 & \lambda-5 & 0\\ 0 & 0 & 0 & 0 & 0\end{pmatrix} = \begin{pmatrix}\mathbf{a}_1\\ \mathbf{a}_2'\\ \mathbf{a}_3'\\ \mathbf{a}_4'\\ \mathbf{0}\end{pmatrix}$$

В резултат получаваме следните два случая:

1) ако $\lambda \neq 5$ следва, че $r(A) = 4$;
2) ако $\lambda = 5$ следва, че $r(A) = 3$.

б) При $\lambda = 5$ имаме, че $\dim W = 3$, тъй като

$$W = \ell(\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_4, \mathbf{a}_5) = \{w = \mu_1\mathbf{a}_1 + \mu_2\mathbf{a}_2' + \mu_3\mathbf{a}_3' \mid \mu_i \in \mathbb{R}, i = 1,2,3\}.$$

Образуваме хомогенната система линейни уравнения (ХСЛУ) с коефициенти координатите на вектор-редовете a_1, a_2' и a_3' на матрицата A:

$$U: \left|\begin{array}{rcrcrcrcl} x_1 & - & 5x_2 & + & 17x_3 & & & = & 0\\ & & x_2 & - & 10x_3 & - & x_4 & = & 0\\ & & & & 15x_3 & + & 2x_4 & - & x_5 = 0\end{array}\right. . \tag{1}$$

Общият вид на решението на системата (1) има вида:

$$(33p + 5q, 10p + q, p, q, 15p + 2q),$$

а една нейна фундаментална система от решения (ФСР) е
$\mathbf{u}_1 = (33, 10, 1, 0, 15)$, $\mathbf{u}_2 = (5, 1, 0, 1, 2)$, т.е.

$$U = \{u = \alpha_1\mathbf{u}_1 + \alpha_2\mathbf{u}_2 \mid \alpha_i \in \mathbb{R}, i = 1,2\}$$

и в частност $\dim U = 2$.

Тогава търсената ХСЛУ, такава че пространството от решенията ѝ съвпада с подпространството $W \leq \mathbb{R}^5$ е:

$$W: \left|\begin{array}{rcrcrcrcl} 33x_1 & + & 10x_2 & + & x_3 & & + & 15x_5 & = 0\\ 5x_1 & + & x_2 & & & + & x_4 & + 2x_5 & = 0\end{array}\right. . \tag{2}$$

в) От условие б) имаме, че ХСЛУ (2) задава подпространството W. Тогава векторът $\mathbf{a} = (1, \alpha, 12, \beta, 1) \in W \Leftrightarrow$

$$\left|\begin{array}{rcrcrcrcl} 33 & + & 10\alpha & + & 12 & + & 15 & = 0\\ 5 & + & \alpha & + & \beta & + & 2 & = 0\end{array}\right. \Leftrightarrow \left|\begin{array}{rcr} \alpha & = & -6\\ \beta & = & -1\end{array}\right. .$$ Следователно $\mathbf{a} = (1, -6, 12, -1, 1) \in W$.

Критерии за оценяване

а) **3 т.** при вярно и пълно решение;

б) **4 т.** при вярно и пълно решение;

в) **3 т.** при вярно и пълно решение.

Задача 4. Даден е следният псевдокод:

```
1 Program BestInterest
2 Interest, Base Rate, Balance: Real
3
4 Begin
5 Base Rate = 0.035
6 Interest = Base Rate
7
8 Read (Balance)
9 If Balance > 1000
10 Then
11  Interest = Interest + 0.005
12  If Balance < 10000
13  Then
14  Interest = Interest + 0.005
15  Else
16  Interest = Interest + 0.010
17  Endif
18 Endif
19
20 Balance = Balance × (1 + Interest)
21
22 End
```

Да се конструира модел за тестване с граф на управляващия поток, от който след това да се дефинират тестови сценарии за структурно тестване (тестване по метода на бялата кутия).

Да се опишат основните стъпки при конструиране на модела, като се посочат номерата на редовете, които се вземат предвид.

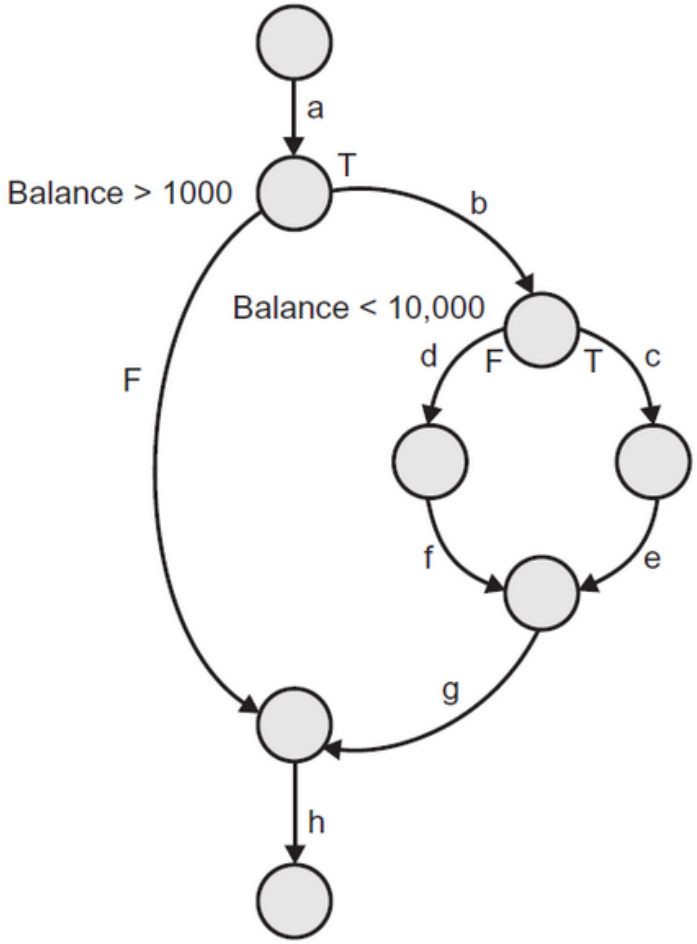
а) Колко процента е покритието на условните оператори при следния тестов сценарий: `Balance=900`?

б) Да се дефинират тестови сценарии за постигане на 100% покритие на изразите.

в) Да се дефинират тестови сценарии за постигане на 100% покритие на условните оператори

г) Колко тестови сценария са необходими за да се постигне 100% покритие на изразите и условните оператори?

Примерно решение



Последователност на конструиране на граф на управляващия поток

- асоцииране на обработващите възли с изразите за присвояване, извикване на процедури или функции;
- асоцииране на възлите за взимане на решение с изразите за условен преход „if-then-else“ или „if-then“;
- създаване на специален тип възли за разклонение и асоциирането им с изразите за цикъл;
- асоцииране на началния и крайния възел на графа с първия и последния израз в програмата.

а) 30%

б) Очаква се дефиниране на два тестови сценария за постигане на 100% покритие на изразите:
TC1: Balance = 5 000
TC2: Balance = 10 000

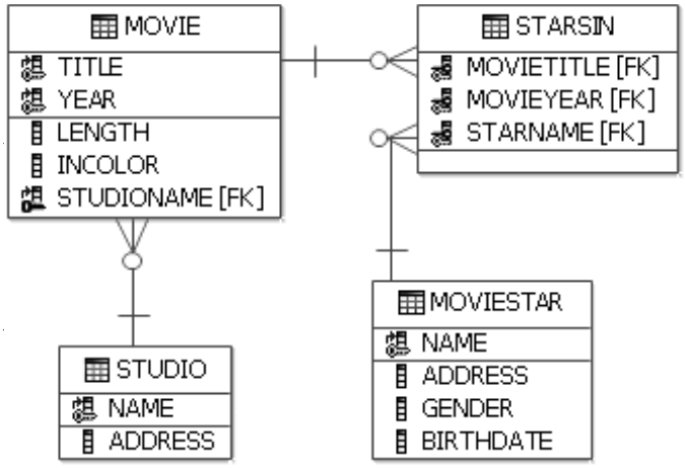
в) Очаква се дефиниране на 3 тестови сценарии за постигане на 100% покритие на условните оператори:
TC1: Balance = 5 000
TC2: Balance = 10 000
TC3: Balance = 800

г) 3

Критерии за оценяване

- Модел за тестване с граф на управляващия поток: 3 т.
- Описани стъпки за конструиране на модела: 2 т.
- Определен процент на покритие на примерния тестов сценарий: 0,5 т.
- Дефинирани тестови сценарии за 100% покритие на изразите: 2 т.
- Дефинирани тестови сценарии за 100% покритие на условните оператори: 2 т.
- Правилно определен брой тестови сценарии за постигне 100% покритие на изразите и условните оператори: 0,5 т.

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



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`.

Таблицата `MovieStar` съдържа информация за филмови звезди:

- `name` — име, първичен ключ;
- `address` — адрес;
- `gender` — пол, `'M'` за мъж (актьор) и `'F'` за жена (актриса);
- `birthdate` — рождена дата.

Таблицата `StarsIn` съдържа информация за участието на филмовите звезди във филмите. Трите атрибута заедно формират първичния ключ. Атрибутите `movietitle` и `movieyear` образуват външен ключ към `Movie`.

- `movietitle` — заглавие на филма;
- `movieyear` — година на заснемане на филма;
- `starname` — име на филмовата звезда, външен ключ към `MovieStar.name`.

1) Да се напише заявка, която извежда без повторение имената на всички актриси, които са играли във филми както през 20-и век (до 2000 г. включително), така и през 21-и век (от 2001 г. насам).

2) Да се посочи коя от следните заявки извежда без повторение имената на всички студиа, на които сумарната дължина на всички техни черно-бели филми е по-голяма от дължината на най-дългия филм на това студио (без значение дали е цветен):

А)
```
SELECT name
FROM Studio
LEFT JOIN Movie
     ON name = studioname
     AND incolor = 'N'
GROUP BY name
HAVING SUM(length) > MAX(length);
```

Б)
```
SELECT DISTINCT name
FROM Studio
RIGHT JOIN Movie
      ON name = studioname
      AND incolor = 'N'
WHERE SUM(length) > MAX(length);
```

В)
```
SELECT name
FROM Studio
WHERE (SELECT SUM(length)
       FROM Movie
       WHERE incolor = 'N'
       AND studioname = Studio.name)
   >  (SELECT MAX(length)
       FROM Movie
       WHERE studioname = Studio.name);
```

Г)
```
SELECT DISTINCT studioname
FROM Movie m
WHERE (SELECT SUM(length)
       FROM Movie
       WHERE incolor = 'N'
         AND studioname = m.studioname)
    > (SELECT MAX(length)
       FROM Movie);
```

Примерно решение на подзадача 1:

```
SELECT starname
FROM StarsIn
JOIN MovieStar ON starname = name
WHERE movieyear <= 2000 AND gender = 'F'
INTERSECT
SELECT starname
FROM StarsIn
JOIN MovieStar ON starname = name
WHERE movieyear > 2000 AND gender = 'F';
```

Критерии за оценяване:

1) Общо 5 т. за изцяло коректно решение, като:
   - точките се намаляват с 3 при комбиниране на двете условия за годините по такъв начин, че да се извеждат всички актриси, които са играли в произволен филм (например, вариант на примерното решение с използване на `UNION`, вместо `INTERSECT`);
   - за всяка сгрешена клауза броят на точките се намалява с 2 до достигане на 0 т.

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. Да се направи декомпозиция на модулите от архитектурата на софтуерна система за управление на банкови профили според дадените по-долу изисквания. Да се обоснове защо така проектираната архитектура удовлетворява изискванията.

R1. Системата работи като мобилно приложение за управление на банкови сметки, което комуникира с централен сървър на банката.

R2. Всеки клиент на банката при изтегляне на приложението трябва да се регистрира използвайки номера на сметката си и уникален идентификатор, който се взема лично от клон на банката (т.нар. токен).

R3. Чрез приложението всеки клиент на банката да може да изпраща и получава суми използвайки телефонния номер на друг клиент, който има инсталирано съответното приложение.

R4. Чрез приложението всеки клиент на банката да може да тегли пари от сметката си и да внася пари в нея чрез банкомат на банката като използва телефона си. За целта през приложението се генерира уникален код, който се въвежда на банкомата за идентифициране на клиента.

R5. Всеки клиент на банката да може да плаща в магазини, които са потребители на банката, използвайки QR код, свързан с банковата сметка на магазина.

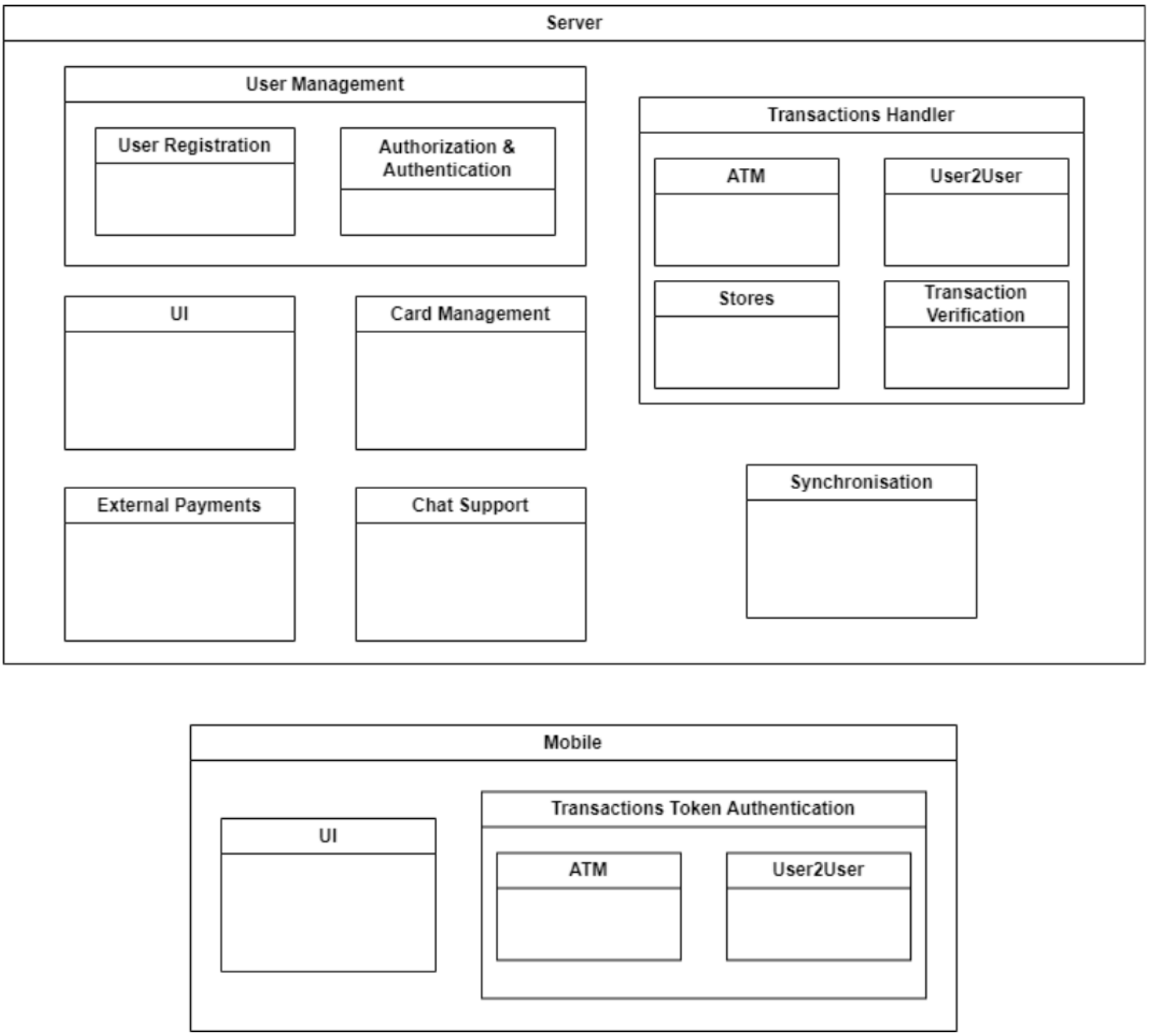
R6. Чрез приложението да има възможност за връзка чрез чат с отдел за поддръжка на банката.

R7. Всеки клиент на банката да може да управлява множество от своите карти.

R8. Всеки клиент на банката да може да управлява плащането на различни типове сметки (данъци, ток, вода и др.) през приложението. За целта трябва да се предвиди връзка с информационните системи на съответните оператори на тези услуги.

R9. Системата трябва да е достъпна 24/7.

R10. Данните на клиентите на банката трябва да са защитени от външна намеса.



Фигура 1: Декомпозиция на модулите

Примерно решение

Декомпозицията на модулите на системата е представена на фиг. 1. Според условието на задачата, системата трябва да работи като мобилно приложение и затова се разделя на две подсистеми – мобилно приложение (Mobile) и сървърен модул (Server). Потребителският интерфейс на системата е реализиран в модула UI.

За съхраняване на данните от потребителските профили и другите чувствителни данни, се използва база данни, информацията в която се съхранява в криптиран вид. Също така, за удовлетворяването на изискване R10 се налага всички комуникации да са криптирани и да се използват сигурни протоколи за осъществяването им (напр. HTTPS). С цел преносимост, връзката с базата данни се осъществява чрез модула DB Connector. Модулът User Manager осигурява функционалност за регистрация на потребителите (User Registration) и управлява политиките за достъп (Authorization & Authentication). С това е изпълнено изискване R2.

Три от основните функции на системата, свързани с транзакции чрез мобилното устройство, се осъществяват в модул Transactions Handler в сървъра. Подмодулът Transaction Verification се грижи за създаването на уникалните кодове, нужни за транзакциите чрез банкомат и между два клиента на банката (модули User2User и ATM). Съответно в модулът Transactions Token Authentication (в Mobile) се проверяват кодовете за осъществяването на транзакцията. Модулът Stores се грижи за плащанията чрез QR кодове изпратени от клиента. С това са удовлетворени изисквания R3, R4 и R5.

В модул Chat Support е предвидено да се реализира функционалността за чат с отдела за поддръжка на банката. Така се удовлетворява изискване R6.

Модул Card Management се грижи за управлението на различните карти на всеки клиент. От там могат да си следят транзакциите и да активират/деактивират техните карти и с това е удовлетворено изискване R7.

Модулът External Payments служи като междинно ниво, чрез което се осъществяват връзките с различните външни системи за плащането на сметки и така е удовлетворено изискване R8.

Изискване R9 може да се удовлетвори като се направи репликация (излишък) на сървъра и базата данни, като модулът Synchronization има задача да осигури консистентност на данните между различните активни копия на системата (репликацията не е показана на фиг. 1, тъй като се счита, че тя не е предмет на декомпозицията на модулите).

Критерии за оценяване

Удовлетворено е изискването за транзакции между клиенти на банката чрез мобилно устройство (R3)	2 т.
Удовлетворено е изискването за транзакции на банкомат използвайки мобилно устройство (R4)	2 т.
Удовлетворено е изискването за плащане чрез QR код (R5)	1 т.
Удовлетворено е изискването за чат с отдел за поддръжка на банката (R6)	1 т.
Удовлетворени са изискванията за управление на карти и плащане на сметки (R7, R8)	1 т.
Удовлетворени са изискванията за сигурност (R10)	1 т.
Удовлетворени са изискванията за наличност в системата (R9)	2 т.

Задача 7. Вашата фирма е сключила договор за създаване на национална облачна среда за обучение на учениците и осъществяване на дигитална трансформация на образованието в България.

Да се подготви план за работа със заинтересованите лица:

1 Да се идентифицират всички заинтересованите лица (организации и частни лица), които са засегнати от проекта (вътрешни и външни), като се определят техните интереси, участие и въздействие върху успеха на проекта.

2 За всяка група заинтересовани лица да се определи в какви проектни дейности (задачи) трябва да бъдат включени.

Примерно решение

1 Заинтересовани лица:

- **вътрешни:**
  - i бизнес анализатори, образователни експерти – проучват теорията и практиката в образованието и подготвят изискванията към облачната среда;
  - ii софтуерни специалисти (вкл. архитекти, програмисти, тестери...) – проектират и разработват софтуера;
  - iii хардуерни специалисти (компютри, мрежи...) – проектират и разработват облачната среда мениджъри и финансисти – ръководят и контролират проектните дейности (план, график, бюджет, качество, рискове...) и ги съгласуват с възложителя;
  - iv обучители – обучават учители и директори за разработване на дигитално съдържание и ползване на облачната среда;
- **външни:**
  - i министерство на образованието и науката – възложител, контролира, валидира и съгласува с други министерства и ведомства дейности по проекта;
  - ii регионални инспекторати по образование – съдействат за организиране на проектни дейности в региона им, участват във валидацията на проекта и реализацията на облачната среда и комуникацията с училища, директори и учители;
  - iii училища – участват в разработването на дигитално съдържание и използване на облачната среда;
  - iv НПО на учители – участват във валидацията на облачната среда и дигитализацията на съдържанието и комуникацията с учители;
  - v правителствени експерти – участват в контрола, валидацията и съгласуване с други министерства и ведомства дейности по проекта;
  - vi директори/учители – участват в контрола на разработването на дигитално съдържание и използване на облачната среда в училището им;
  - vii ученици – участват в използване на облачната среда и дигиталното съдържание;
  - viii родители – участват в използване на облачната среда за контрол на децата си.

Критерии за оценяване

1 Идентифицирани са всички външни и вътрешни за проекта заинтересовани лица, като за всяко е посочено какви интереси има в проекта – макс. 6 точки.

- Описание на вътрешни заинтересовани лица с отчитане на различната експертиза (мениджмънт, финансова, техническа) – до 2 т.
- Описание на външни заинтересовани лица – изброяване на типове организации с роля в образованието и физически лица, имащи отношение към училищното обучение – до 4 т.

2 За всяко заинтересовано лице е определена дейност, в която може да участва – макс. 4 точки.

- Оценява се дали е определена роля на всички заинтересовани лица и дали не е пропусната важна проектна дейност.

Забележка: Не се дават точки за втората част от задачата, ако не са определени мястото и ролята на заинтересованите лица в първата част.

Задача 8. Дадени са две урни. В първата има 3 бели и 2 черни топки, а във втората урна има 1 бяла и 2 черни топки. На всеки опит от първата урна се теглят две топки без връщане, а от втората урна се тегли една топка. Дефинирани са следните събития:

$A = \{$от първата урна е изтеглена точно една бяла топка$\}$;

$B = \{$от първата урна са изтеглени повече бели отколкото от втората$\}$.

а) да се определи вероятността на A;

б) да се определи вероятността на B;

в) да се определи вероятността на A, ако знаем че се е изпълнило B;

г) ако се провеждат серия от опити докато събитието B се изпълни, като след всеки опит съставът на урните се възстановява в първоначалния вид, какъв е средният брой на направените опити?

Примерно решение

а) Вероятност на A:

Събитието зависи само от първите две топки, така че третата топка няма значение. Всички начини за избиране на две топки с наредба без повторение са $V_6^2 = 6.5 = 30$. Топките са различни, тогава от съображения за симетрия в половината случаи първата топка ще е по-голяма, т.е. общо в 15 случая.

б) Вероятност на B:

Ще разгледаме две възможности за сбъдване на събитието B.

I случай. Ако от първата урна е изтеглена една бяла топка, т.е. се е изпълнило събитието A, тогава от втората задължително трябва да е изтеглена черна. Вероятността за черна топка от втората урна е $\frac{2}{3}$, тогава вероятността на този случай е $\frac{6}{10}\frac{2}{3} = \frac{4}{10}$.

II случай. Ако от първата урна са изтеглени две бели топки, събитието B се изпълнява без значение какво е станало с втората урна. Вероятността за две бели от първа урна е $\binom{3}{2}/\binom{5}{2} = \frac{3}{10}$.

Съгласно формулата за пълна вероятност:

$$P(B) = \frac{4}{10} + \frac{3}{10} = \frac{7}{10}.$$

в) Вероятността на A, ако знаем че се е изпълнило B:

Ще пресметнем вероятността на сечението, т.е. вероятността A и B да се изпълнят едновременно. Това означава да изтеглим точно една бяла от първата урна и черна от втората, което е I случай на предишното подусловие $P(AB) = \frac{4}{10}$.

За търсената условна вероятност получаваме:

$$P(A|B) = \frac{P(AB)}{P(B)} = \frac{4/10}{7/10} = \frac{4}{7}.$$

г) Ще наричаме сбъдването на събитието B успех. В условието на задачата разпознаваме “Схема на Бернули”. Наистина, съставът на урните не се променя, което означава, че вероятността да се изпълни събитието B на всеки опит е $\frac{7}{10}$ и освен това резултатът от един опит не влияе на резултата на следващите опити, т.е. опитите са независими. Нека X броя на направените опит до първото сбъдване на B. В задачата се търси математическото очакване на X.

Ще означим с Y броя на неуспехите до първия успех. Ясно е, че Y е геометрично разпределена случайна величина $Y \in Ge(p)$ и е известна формулата за математическото очакване $EY = \frac{q}{p}$. В нашия случай $p = \frac{7}{10}$, следователно $EY = \frac{3}{7}$.

Броят на проведените опити, разбира се, е с един повече от броя на неуспехите до първия успех, т.е. $X = Y + 1$ следователно:

$$EX = EY + 1 = \frac{3}{7} + 1 = \frac{10}{7}.$$

Критерии за оценяване

а) **2 т.** при вярно и пълно решение;

б) **2 т.** при вярно и пълно решение;

в) **2 т.** при вярно и пълно решение;

г) **4 т.** при вярно и пълно решение.

Чернова