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++.

Под всеки от дадените по-долу фрагменти да се посочи какво ще изведе той на стандартния изход.

1)

```
void print(char* str) {
  char* ptr = str;
  while (*ptr != '\0') {
      *ptr = *ptr+1;
      ptr++;
  }
  std::cout << str;
}
void main() {
  char str[] = "ACE";
  print(str);
}
```

Отговор: _____

2)

```
void func(int& x, int* y, int z){
  int a = 3;
  int b = 5;
  x = z - b;
  *y = 2 * z;
  z = x ? a + *y : 0;
}
void main() {
  int a = 2; int b = 4;
  func(b, &a, b);
  std::cout << a << endl;
```

Отговор: _____

```
  std::cout << b << endl;
```

Отговор: _____

3)

```
int total = 0;
for (int i = 2; i <= 10; ) {
  if (i % 3 == 0) {
    total = total + i;
  }
  i += 2;
}
std::cout << total << endl;
```

Отговор: _____

4) При какво условие ще спре да се изпълнява цикълът while?

```
while ((a || b) && (c || d)) {...}
```

а) ако `!(a && b)` е истина

б) ако `!a || !b || !c || !d` е истина

в) ако `(!a && !b) || (!c && !d)` е истина

г) ако `(!a || !b) && (!c || !d)` е истина

д) нито един от изброените

Отговор: _____

5) Да се попълнят фрагментите така, че функцията `reverseMatrixRows` да обръща местата на редовете на `mat` относно центъра (т.е. първия ред да се размени с последния и т.н.), а функцията `reverseMatrixCols` да обръща местата на колоните на матрицата `mat` относно центъра. Нека `r` и `c` са съответно брой редове и колони в матрицата.

```
void reverseMatrixRows(
       int** mat, int r, int c) {
  for (int i = 0; i < _____; ++i)
    for (int j = 0; j < c; ++j)
     swap(mat[i][j], _____________);
}

void reverseMatrixCols(
        int** mat, int r, int c) {
  for (int i = 0; i < r; ++i)
   for (int j = 0; j < _____; ++j)
     swap(mat[i][j], _____________);
}
```

6) Да се попълнят празните места в рекурсивната функция така, че `revStr` да обръща елементите на подаден като параметър низ `str`, които се намират на индекси в интервала `[start, end]`;

```
void revStr(_____________ str,
  unsigned start, unsigned end) {
  if (_________________) {
    return;
  }
  swap(str[start], str[end]);
  revStr(str, ________,_________);
}
```

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 довършване на кода на функциите, 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. Изцяло правилно попълнен отговор носи 1 точка.
2. Всяко коректно попълнено празно място носи по 1 точка (общо 2 точки).
3. Изцяло правилно попълнен отговор носи 1 точка.
4. Изцяло правилно попълнен отговор носи 1 точка.
5. Всяко коректно попълнено празно място носи по 0.5 точка (общо 2 точки).
6. Всеки изцяло коректно попълнен празен ред на рекурсивната функция носи по 1 точка (общо 3 точки).

Примерно решение:

1) Изцяло правилно попълнен отговор носи 1 точка.
Отговор: BDF

2) Всяко коректно попълнено празно място носи по 1 точка (общо 2 точки).
Отговор: 8
Отговор: -1

3) Изцяло правилно попълнен отговор носи 1 точка.
Отговор: 6

4) Изцяло правилно попълнен отговор носи 1 точка.
Отговор: в)

5) Всяко коректно попълнено празно място носи по 0.5 точка (общо 2 точки).

```
void reverseMatrixRows(
       int** mat, int r, int c) {
  for (int i = 0; i < r / 2; ++i)
    for (int j = 0; j < c; ++j)
     swap(mat[i][j], mat[r-1-i][j]);
}

void reverseMatrixCols(
        int** mat, int r, int c) {
  for (int i = 0; i < r; ++i)
   for (int j = 0; j < c / 2; ++j)
     swap(mat[i][j], mat[i][c-1-j]);
}
```

6) Всеки изцяло коректно попълнен празен ред на рекурсивната функция носи по 1 точка (общо 3 точки).
Забележка: Използването на char е също допустимо решение и носи пълен брой точки.*

```
void revStr( string& str,
  unsigned start, unsigned end) {
  if (start >= end) {
    return;
  }

  swap(str[start], str[end]);

  revStr(str, start+1, end-1);
}
```

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

1) Разгледайте дадената по-долу програма. В полето за отговор напишете какво ще изведе тя.

```
1  class A {
2  public:
3    A()  { prn('c'); }
4    ~A() { prn('d'); }
5    virtual void prn(char ch) const {
6      std::cout << 'A' << ch << ';';
7    }
8  };
9
10 class B : public A {
11 public:
12   B()  { prn('c'); }
13   ~B() { prn('d'); }
14   virtual void prn(char ch) const {
15     std::cout << 'B' << ch << ';';
16   }
17 };
18
19 int main() {
20   A* obj = new B;
21   obj->prn('p');
22   delete obj;
23 }
```

Отговор: _____

2) Срещу всяко от твърденията напишете „Да“, ако то е вярно, или „Не“, ако е невярно.

А) Конструкторът на един клас може да бъде статичен.	
Б) Деструкторът на един клас може да бъде статичен.	
В) В статична член-функция на даден клас можем да достъпваме както статичните, така и нестатичните член-променливи на класа.	
Г) В статична член-функция на даден клас можем да направим обръщение към виртуална член-функция на същия клас без да има създаден обект.	
Д) Във виртуална функция на даден клас можем да направим обръщение към статична член-функция на същия клас.	
Е) Ако в един абстрактен клас има член-функция, тя задължително трябва да бъде виртуална.	

3) Довършете програмата (на подчертаните места) така, че да бъде синтактично коректна и да спазва принципите на ООП в C++.

```
1  class Item {
2    std::string m_label;
3  public:
4    Item(std::string t) : m_label(t) {}
5
6    ___________ ~Item() {}
7
8    const std::string& label() const {
9      return m_label;
10   }
11 };
12
13 _________________ <typename T>
14 class Box : public Item{
15   _______ m_contents;
16 public:
17   Box(std::string label, const T& contents)
18     : Item(_______________)
19   {
20     m_contents = new T(___________________);
21   }
22
23   Box(const Box&) = delete;
24   Box& operator=(const Box&) = delete;
25   ___________ ~Box() { delete m_contents; }
26   T& contents() { return _________________; }
27 };
28
29 Box<int>* createRandomIntBox() {
30   int randomValue = rand() % 100 + 1;
31   return ___________________________("
       random box", randomValue);
32 }
33
34 Box<int>* toIntBox(Item* p) {
35   return dynamic_cast<Box<int>*>(p);
36 }
37
38 int main() {
39   Item* p = createRandomIntBox();
40   toIntBox(p)->contents() = 42;
41   std::cout << "Box {"
42             << "\n  Label: " << p->label()
43             << "\n  Contents: "
44             << toIntBox(p)->contents()
45             << "\n}\n";
46   delete p;
47 }
```

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, получени като сума от дадените подолу компоненти, но само ако са посочени точно те и точно в правилния ред.

- 1 точка, ако като първо е посочено `Ac;Bc;`. Ако редът е объркан, има липсващи символи и т.н. се оценява с 0 точки.
- 1 точка, ако като второ е посочено `Bp;`.
- 1 точка, ако накрая е посочено `Ad;`.
- Ако в отговора не са поставени точките и запетаите, точките се намаляват с 0,5.
- Ако редът на последователностите е объркан, точките се намаляват наполовина.
- Всеки излишен елемент намалява точките с 1.

2) 3 точки (по 0,5 за всеки верен отговор).

3) 4 точки (по 0,5 за всеки верен отговор).

Примерно решение

1) `Ac;Bc;Bp;Ad;`

2)

А)	Не
Б)	Не
В)	Не
Г)	Не
Д)	Да
Е)	Не

3)

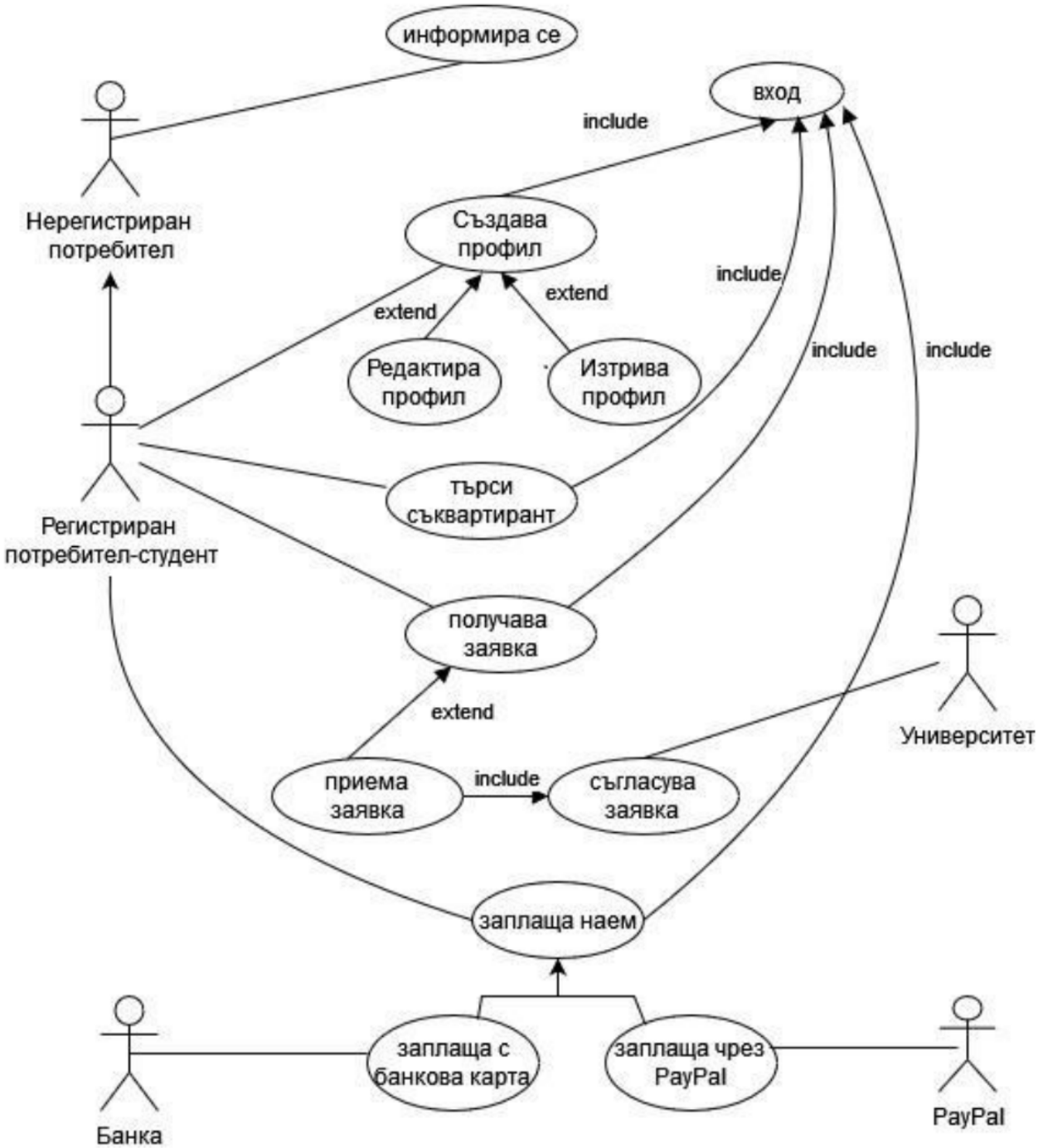
```
1  class Item {
2    std::string m_label;
3  public:
4    Item(std::string t) : m_label(t) {}
5
6    virtual ~Item() {}
7
8    const std::string& label() const {
9      return m_label;
10   }
11 };
12
13 template <typename T>
14 class Box : public Item{
15   T* m_contents;
16 public:
17   Box(std::string label, const T& contents)
18     : Item(label)
19   {
20     m_contents = new T(contents);
21   }
22
23   Box(const Box&) = delete;
24   Box& operator=(const Box&) = delete;
25   virtual ~Box() { delete m_contents; }
26   T& contents() { return *m_contents; }
27 };
28
29 Box<int>* createRandomIntBox() {
30   int randomValue = rand() % 100 + 1;
31   return new Box<int>("random box",
       randomValue);
32 }
33
34 Box<int>* toIntBox(Item* p) {
35   return dynamic_cast<Box<int>*>(p);
36 }
37
38 int main() {
39   Item* p = createRandomIntBox();
40   toIntBox(p)->contents() = 42;
41   std::cout << "Box {"
42             << "\n  Label: " << p->label()
43             << "\n  Contents: "
44             << toIntBox(p)->contents()
45             << "\n}\n";
46   delete p;
47 }
```

Бележка: На ред 25 се получават пълни точки и ако полето е оставено празно, но само ако деструкторът на `Item` е посочен като `virtual`.

Задача 3. Да се състави диаграма на потребителските случаи на следните изисквания на софтуерно приложение, което подпомага студенти, ползващи общежитие, да открият подходящ съквартирант:

1. Софтуерното приложение предоставя на потребителите информация за наличните общежития.
2. Регистрирани потребители-студенти могат да създават, редактират и изтриват профила си с интереси и хобита.
3. Регистрирани потребители-студенти могат да търсят съквартирант по интереси и хобита.
4. Регистрирани потребители-студенти могат да получават заявка от друг студент, желаещ да бъде техен съквартирант и евентуално да приемат заявката. Окончателното одобрение на заявката става след съгласуване с Университета.
5. Регистрирани потребители-студенти могат да заплащат месечния си наем за общежитие онлайн с банкова карта или чрез системата PayPal.

Примерно решение



Критерии за оценяване

А) За представени всички актьори – 4 точки. За всеки неотбелязан актьор точките се намаляват пропорционално;

Б) За представени всички случаи на употреба – 4 точки. За неотбелязан случай на употреба точките се намаляват пропорционално;

В) За записани вярно всички връзки – 2 точки. За неотбелязана или грешна връзка точките се намаляват пропорционално.

За оформяне на крайната оценка сумата от точките се закръглява до цяло число.

Задача 4.
Онлайн магазин предлага отстъпки на своите клиенти, както следва:

- Ако клиентът е лоялен, получава 5% отстъпка.
- Ако поръчката е над 100 лв., се добавя 10% отстъпка, независимо дали клиентът е лоялен.
- Ако е налична текуща промоция, се добавя 5% отстъпка.
- Ако клиентът разполага с промо код, се добавя 5% отстъпка, но само при условие, че не е налична текуща промоция.

Да се дефинира таблица за вземане на решение, въз основа на която могат да се генерират тестови сценарии. Таблицата трябва да показва условията, следствията от тях и правилата, от които се генерират тестовите сценарии.

Въз основа на таблицата за вземане на решение да се определи резултата от изпълнението на следните тестови сценарии:

- TC1: Лоялен клиент с промо код, поръчка над 100 лв. и наличие текуща промоция.
- TC2: Регулярен клиент без промо код, с поръчка под 100 лв. и наличие текуща промоция.

Критерии за оценяване

- Дефинирани условия и следствия: 1 т.
- Дефинирани стъпки за създаване на таблица за вземане на решения: 3 т.
- Създадена таблица за вземане на решения: 4 т.
- Редуциране на правила: 1 т.
- Определяне на резултати от примерните тестови сценарии: 1 т.

Примерно решение

Стъпка 1: Идентифицират се следните условия и следствия

- C1: Лоялен клиент
- C2: Поръчка над 100 лв.
- C3: Налична текуща промоция
- C4: Клиент с промо код
- E1: Отстъпка 20%
- E2: Отстъпка 15%
- E3: Отстъпка 10%
- E4: Отстъпка 5%

Стъпка 2: Условията и следствията се изброяват в таблица на решенията.
Стъпка 3: Броят на комбинациите е $2^3 = 8$.
Стъпка 4: Колоните на таблицата се попълват като се има предвид, че:

- $RF1 = RF2 = 8/2 = 4$
- $RF3 = 4/2 = 2$
- $RF4 = 2/2 = 1$

Стъпка 5: Определяне на следствията от всяко правило.
Стъпка 6: Определя се очаквания резултат от изпълнение на всяко следствие:

Условия	**Стойности**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
C_1	T, F	T	T	T	T	T	T	T	T	F	F	F	F	F	F	F	F
C_2	T, F	T	T	T	T	F	F	F	F	T	T	T	T	F	F	F	F
C_3	T, F	T	T	F	F	T	T	F	F	T	T	F	F	T	T	F	F
C_4	T, F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F
Действия																	
E_1	20%	T	T	T													
E_2	15%				T					T	T	T					
E_3	10%					T	T	T					T				
E_4	5%								T					T	T	T	
Checksum	16	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

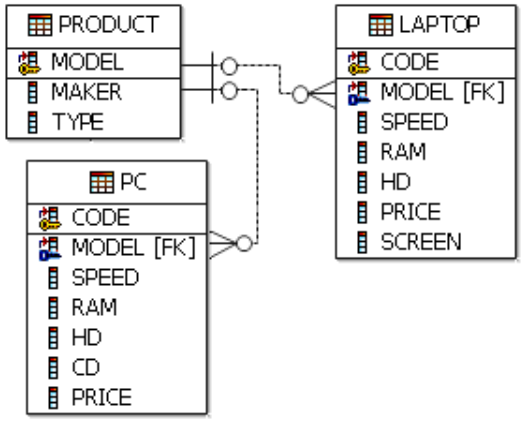
Стъпка 7: Редуцират се правилата:

Условия	**Стойности**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
C_1	T, F, N/A	T	T	T	T	T	T	F	F	F	F	F	F	F
C_2	T, F, N/A	T	T	T	F	F	F	T	T	T	F	F	F	F
C_3	T, F, N/A	T	F	F	T	F	F	T	F	F	T	T	F	F
C_4		N/A	T	F	N/A	T	F	N/A	T	F	T	F	T	F
Действия														
E_1	20%	T	T											
E_2	15%			T				T	T					
E_3	10%				T	T				T				
E_4	5%						T				T	T	T	
Checksum	16	2	1	1	2	1	1	2	1	1	1	1	1	1

Стъпка 8: Генерират се тестови сценарии, съответстващи на правилата в таблицата:

- Резултат TC1: 20%.
- Резултат TC2: 5%.

Задача 5. Дадена е базата от данни PC. В нея се съхранява информация за два вида продукти – лаптопи и персонални компютри.



Таблицата `Product` съдържа базова информация за всеки продукт:

- `model` — модел на продукта, първичен ключ;
- `maker` — производител на продукта;
- `type` — един от типовете: `'PC'`, `'Laptop'`.

Таблицата `PC` съдържа специфична информация за персоналните компютри:

- `code` — уникален идентификатор на дадена компютърна конфигурация, първичен ключ;
- `model` — модел на компютъра, външен ключ към `Product.model`. Може да имаме няколко различни компютърни конфигурации от един и същ модел, но с различни параметри. Може да бъде неизвестен (`NULL`);
- `speed` — тактова честота на процесора в GHz;
- `ram` — количество RAM памет в GB;
- `hd` — размер на твърдия диск в TB;
- `cd` — скорост на компактдисковото устройство;
- `price` — цена на компютъра.

Таблицата `Laptop` съдържа специфична информация за лаптопите. Атрибутите са аналогични на тези на PC, но липсва атрибутът `cd` и има атрибут `screen`, указващ диагонала на екрана в инчове.

1) Да се напише заявка, която намира средната цена на всички лаптопи, удовлетворяващи поне едно от следните условия:

- RAM паметта им е между 16 и 32 GB включително;
- производителят е неизвестен (колоната maker има стойност `NULL`).

Заявката да изведе точно една стойност. Ако не съществува нито един лаптоп, удовлетворяващ някое от условията, заявката да изведе стойността `NULL`.

2) Коя от следните заявки извежда информация колко различни модела 14-инчови лаптопи произвежда всеки от производителите? В резултатната таблица трябва да бъдат изведени без повторение абсолютно всички производители без изключение, дори и да нямат лаптопи изобщо. От даден модел лаптопи може да има няколко различни конфигурации – в резултата трябва да бъде третиран като един модел.

```
А) SELECT DISTINCT maker,
     (SELECT COUNT(*)
      FROM Laptop
      WHERE screen = 14)
   FROM Product p;
```

```
Б) SELECT maker, COUNT(DISTINCT Laptop.model)
   FROM Laptop
   RIGHT JOIN Product
         ON Laptop.model = Product.model
            AND screen = 14
   GROUP BY maker;
```

```
В) SELECT DISTINCT maker, COUNT(Laptop14.model)
   FROM Product
   LEFT JOIN (SELECT model, screen
              FROM Laptop) Laptop14
        ON Product.model = Laptop14.model
   GROUP BY maker
   HAVING screen = 14;
```

```
Г) SELECT maker, COUNT(DISTINCT Laptop.model)
   FROM Product
   LEFT JOIN Laptop
        ON Product.model = Laptop.model
   WHERE Laptop.screen = 14
   GROUP BY maker;
```

Примерно решение на подзадача 1:

```
SELECT AVG(price)
FROM Laptop
JOIN Product ON Laptop.model = Product.model
WHERE ram BETWEEN 16 AND 32 OR maker IS NULL;
```

Критерии за оценяване:

1) Коректно решение носи 5 т. За всяка сгрешена клауза броят на точките се намалява с 2 до достигане на 0 т.

2) Единственият верен отговор е Б). 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 онлайн софтуерна система за *наемане на автомобили*. Да се обоснове защо така проектираната архитектура удовлетворява изискванията.

R1. Потребителите трябва да могат да създават акаунти и да влизат с потребителско име и парола. Системата има два вида потребителски профили:

- Обикновени потребители, които използват системата.
- Администратори, които следят за изправността на системата.

R2. Администраторите имат отделен интерфейс за управление на обяви на автомобили, потребителски акаунти и разрешаване на конфликти при резервации.

R3. Администраторите могат да добавят/премахват данни за автомобилите в системата.

R4. Системата трябва да показва списък с налични автомобили с подробности като марка, модел, година, цена и дати на наличност.

R5. Потребителите трябва да могат да търсят автомобили въз основа на различни критерии като местоположение, цена и характеристики на автомобила.

R6. Потребителите трябва да могат да резервират автомобил за конкретни дати и часове.

R7. Потребителите трябва да получават известия (в рамките на 30 мин. след възникване на съответното събитие), за потвърждения на резервации, напомняния, и т.н. Тези известия да може да се получават по имейл, чрез SMS или известие за приложения от трета страна като WhatsApp, Viber и др. Да има възможност да се добавят и други методи за известия към потребителите.

R8. Системата трябва да поддържа сигурна обработка на плащания за такси на резервация.

R9. Системата трябва да поддържа регистър на всички направени резервации и транзакции.

R10. Системата трябва да е на разположение 24/7 с минимално време за престой за поддръжка.

Примерно решение

Декомпозицията на модулите на системата е представена на фиг. 1. Потребителският интерфейс на системата е реализиран в модула UI и е разделен на две части – за потребители (модули Desktop и Mobile) и администратори (модул Admin).

Модул User Handling отговаря за достъпа на потребителите. Съставен е от два под-модула, Register и Login, чрез които се манипулират потребителските профили. Чрез тях се осъществяват регистрацията и удостоверяването и оторизацията (роли и т.н.) при влизане в системата. Те комуникират с базата данни, където е съхранена информацията, която е криптирана, както и всички комуникации (R1).

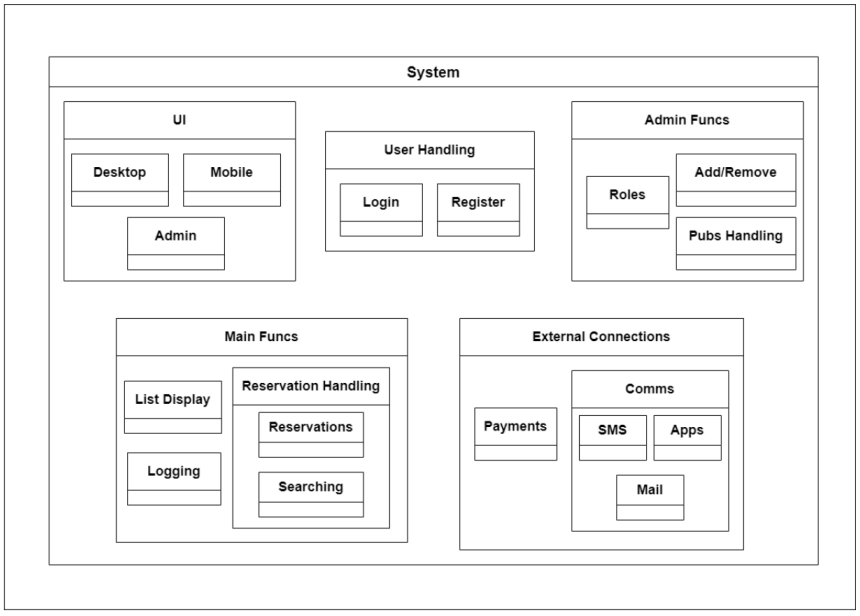
Модул Admin Funcs се грижи за функционалностите на администраторите, чрез трите под-модула – Roles, Add/Remove и Pubs Handling. От Roles се манипулират правата на различните потребители, а от Pubs Handling се манипулират обявите и се решават евентуални конфликти относно обявите (R2). Add/Remove предоставя възможност за добавяне, премахване и модифициране на данните на автомобили (R3).

За основните функционалности на потребителите се грижат модул Main Funcs и неговите под-модули. В модула List Display се показват всички налични обяви за автомобили с всички подробности и опции за съответните потребители (R4).

Функционалностите за търсене и наемане на автомобил са реализирани в модула Reservation Handling. Чрез под-модула Searching, потребителят има възможности за разширено търсене и персонализирани предложения (напр. на базата на това къде се намира). Под-модулът Reservations се грижи за успешното записване в системата на заявката за резервация и (евентуално) плащането, комуникирайки и с други модули, като например Payments (R5, R6) . Последният под-модул на Main Funcs – Logging, се грижи за записването на всички действия в базата и поддържането и актуализирането на регистъра (R9).

Модул External Connections се грижи за комуникацията с външни системи за осъществяването на плащанията и изпращането на известия към потребителите. Чрез Payments системата обработва плащанията, а чрез Comms – нотификациите за различните действия. Вторият модул предоставя и три под-модула – SMS, Apps и Mail. По този начин се свързва с различните системи чрез отделен под-модул и се улеснява бъдещото разширение с други такива системи за комуникация (R7, R8).

Чрез репликация на сървъра и базата данни и използването на подходящи процеси (напр. Load balancing и др.) се осигурява изискването относно наличността (R10). Репликацията не е показана на фиг. 1, тъй като се счита, че тя не е предмет на декомпозицията на модулите.



Фигура 1: Декомпозиция на модулите

Критерии за оценяване

Удовлетворено е изискването за потребителски и администраторски интерфейс (R1, R2)	2 т.
Удовлетворено е изискването за администраторските дейности (R3)	1 т.
Удовлетворено е изискването за показване на списъка с автомобили (R4)	1 т.
Удовлетворени са изискванията за търсене и резервиране на автомобили (R5, R6)	2 т.
Удовлетворено е изискването за получаване на известия (R7)	1 т.
Удовлетворено е изискването за плащания през системата (R8)	1 т.
Удовлетворено е изискването за поддържане на регистър (R9)	1 т.
Удовлетворени са изискванията за наличност (availability) (R10)	1 т.

Задача 7.

Дадена е система линейни уравнения с реални коефициенти, където λ е реален параметър и свободните коефициенти са реални числа:

$$\left|\begin{array}{rrrrl} \lambda x_1+ & x_2+ & x_3+ & x_4 & = b_1 \\ x_1+ & \lambda x_2+ & x_3+ & x_4 & = b_2 \\ x_1+ & x_2+ & \lambda x_3+ & x_4 & = b_3 \\ x_1+ & x_2+ & x_3+ & \lambda x_4 & = b_4 \end{array}\right. \tag{1}$$

а) Да се намери рангът на матрицата на системата в зависимост от стойностите на реалния параметър λ.

б) Да се определи за кои стойности на параметъра λ, системата (1) е съвместима, ако стълбът от свободни членове е $b = (b_1, b_2, b_3, b_4) = (1, \lambda, \lambda, \lambda)$.

в) Да се реши хомогенната система (т. е. $b_1 = b_2 = b_3 = b_4 = 0$) в зависимост от стойностите на реалния параметър λ и да се намери фундаментална система решения, ако $\lambda = -3$.

Примерно решение

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

$$A = \begin{pmatrix} \lambda & 1 & 1 & 1 \\ 1 & \lambda & 1 & 1 \\ 1 & 1 & \lambda & 1 \\ 1 & 1 & 1 & \lambda \end{pmatrix} \sim \begin{pmatrix} 1 & 1 & 1 & \lambda \\ 1 & \lambda & 1 & 1 \\ 1 & 1 & \lambda & 1 \\ \lambda & 1 & 1 & 1 \end{pmatrix} \sim \begin{pmatrix} 1 & 1 & 1 & \lambda \\ 0 & \lambda-1 & 0 & 1-\lambda \\ 0 & 0 & \lambda-1 & 1-\lambda \\ 0 & 1-\lambda & 1-\lambda & 1-\lambda^2 \end{pmatrix}$$

1) Разглеждаме случай $\lambda = 1$. Получаваме

$$A \sim \begin{pmatrix} 1 & 1 & 1 & 1 \end{pmatrix},$$

откъдето $r(A) = 1$.

2) Нека $\lambda \neq 1$. Да разделим втори и трети ред на $(\lambda - 1)$, а четвърти ред на $(1 - \lambda)$, и тогава получаваме:

$$A \sim \begin{pmatrix} 1 & 1 & 1 & \lambda \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \\ 0 & 1 & 1 & 1+\lambda \end{pmatrix} \sim \begin{pmatrix} 1 & 1 & 1 & \lambda \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 1 & \lambda+2 \end{pmatrix} \sim \begin{pmatrix} 1 & 1 & 1 & \lambda \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & \lambda+3 \end{pmatrix}.$$

Ако $\lambda = -3$, то $r(A) = 3$ и в противен случай $r(A) = 4$.

Окончателно,

- ако $\lambda = 1$, то $r(A) = 1$;
- ако $\lambda = -3$, то $r(A) = 3$;
- ако $\lambda \neq 1, -3$, то $r(A) = 4$.

б) От Теоремата на Руше следва, че системата (1) е съвместима тогава и само тогава, когато рангът на матрицата на системата е равен на ранга на разширената матрица на системата, т.е. $r(A) = r(\bar{A})$.

Да разгледаме разширената матрица $\bar{A}$, т.е.

$$\bar{A} = \left(\begin{array}{cccc|c} \lambda & 1 & 1 & 1 & 1 \\ 1 & \lambda & 1 & 1 & \lambda \\ 1 & 1 & \lambda & 1 & \lambda \\ 1 & 1 & 1 & \lambda & \lambda \end{array}\right) \sim \left(\begin{array}{cccc|c} 1 & 1 & 1 & \lambda & \lambda \\ 0 & \lambda-1 & 0 & 1-\lambda & 0 \\ 0 & 0 & \lambda-1 & 1-\lambda & 0 \\ 0 & 1-\lambda & 1-\lambda & 1-\lambda^2 & 1-\lambda^2 \end{array}\right).$$

При $\lambda = 1$, системата има решение. Нека $\lambda \neq 1$, то

$$\bar{A} \sim \left(\begin{array}{cccc|c} 1 & 1 & 1 & \lambda & \lambda \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 1 & 1 & 1+\lambda & 1+\lambda \end{array}\right) \sim \left(\begin{array}{cccc|c} 1 & 1 & 1 & \lambda & \lambda \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & \lambda+3 & \lambda+1 \end{array}\right).$$

Откъдето получаваме, че системата е съвместима, ако $\lambda \neq -3$, тъй като тогава $r(A) = r(\bar{A}) = 4$ и несъвместима в противен случай при $\lambda = -3$, тъй като тогава $r(A) = 3$, а $r(\bar{A}) = 4$.

в) От а) знаем, че системата (1) в зависимост от параметъра λ има следния ранг

- ако $\lambda = 1$, то $r(A) = 1$;
- ако $\lambda = -3$, то $r(A) = 3$;
- ако $\lambda \neq 1, -3$, то $r(A) = 4$.

Разглеждаме случаите:

- При $\lambda = 1$, решаваме системата:

$$\left| x_1 + x_2 + x_3 + x_4 = 0 \right. \tag{2}$$

и решението е $(-p-q-r, p, q, r),\ \forall p, q, r \in \mathbb{R}$.

- При $\lambda = -3$, то $r(A) = 3$ и системата е:

$$\left| \begin{array}{cccccc} x_1 & + & & & - & x_4 = 0 \\ & & x_2 & & - & x_4 = 0 \\ & & & x_3 & - & x_4 = 0 \end{array} \right. \tag{3}$$

и решението е $(p, p, p, p),\ \forall p \in \mathbb{R}$.

- При $\lambda \neq 1, -3$, то $r(A) = 4$ и решението е $(0, 0, 0, 0)$.

Фундаментална система решения, ако $\lambda = -3$ е например $(1, 1, 1, 1)$.

Критерии за оценяване

а) 4 т. при вярно и пълно решение;

б) 3 т. при вярно и пълно решение;

в) 3 т. при вярно и пълно решение.

Задача 8. В урна има 2 бели, 4 зелени и 4 червени топки. Играч залага 1 лв. за правото да изтегли 2 топки от урната. Ако изтегли двете бели топки, играчът печели 5 лв.; ако изтегли една бяла и една зелена, печели 2 лв.; ако изтегли две зелени, печели 1 лв., а във всички останали случаи не печели нищо. Залогът винаги остава за организатора на играта.

а) Каква е вероятността нетно играчът да спечели точно 1 лв. от едно разиграване?

б) Да се намери разпределението на случайната величина „нетна печалба от играта“.

в) Справедлива ли е играта? Справедлива игра е такава, при която математическото очакване на нетната печалба от играта е нула.

г) Разглеждаме като „успех“ игра, при която печалбата е по-голяма от залога (т.е. игра с положителна нетна печалба). Да се намери p – вероятността за „успех“. Нека случайната величина Y е броят на игрите до първия „успех“ включително. Какво е разпределението на Y? Какъв е средният брой игри до първата игра с положителна нетна печалба?

Примерно решение

а) Нетна печалба от 1 лв. имаме в случая, когато печалбата е 2 лв., тъй като залогът е 1 лв. Вероятността за това събитие е: $\frac{\binom{2}{1}\binom{4}{1}}{\binom{10}{2}} = \frac{8}{45}$.

б) Нека с *Net Gain* означим случайната величина "нетна печалба от играта". Тогава разпределението на тази случайна величина е:

$Net\ Gain = x$	4	1	0	-1
$P(Net\ Gain = x)$	1/45	8/45	6/45	30/45

в) Математическото очакване на случайната величина е $E(Net\ Gain) = \sum_x x * P(Net\ Gain = x) = -0.4$ и следователно играта не е справедлива.

г) $p = (Net\ Gain > 0) = (Net\ Gain = 4) + (Net\ Gain = 1) = 1/5$. Разпределението на случайната величина Y е геометрично с параметър $p = 1/5$, т.е.: $Y \sim Ge(p = 1/5)$ или $P(Y = k) = (1-p)^{k-1}p = 0.8^{k-1}0.2$, $k = 1, 2, \ldots$. Очакването на Y е: $EY = 1/p = 5$.

Критерии за оценяване

а) За пълно решение се дава 1 точка.

б) За пълно решение се дават 3 точки, за всяка стойност на случайната величина и нейната вероятност по 1 т. (като се изключва стойността и вероятността от подточка а);

в) За пълно решение се дават 2 точки, за намирането на математическото очакване – 1 т., за извод, че играта не е справедлива – 1 т.;

г) За пълно решение се дават 4 точки, за намирането на p – 1 точка, за намирането на разпределението на Y – 2 точки, за намирането на $E(Y)$ – 1 точка.

Чернова