СОФИЙСКИ УНИВЕРСИТЕТ
“СВ. КЛИМЕНТ ОХРИДСКИ”

ФАКУЛТЕТ ПО МАТЕМАТИКА
И ИНФОРМАТИКА

ДЪРЖАВЕН ИЗПИТ

ЗА ПОЛУЧАВАНЕ НА ОКС “БАКАЛАВЪР ПО КОМПЮТЪРНИ НАУКИ”

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.
- Всяка напълно и коректно решена задача, съгласно критериите за оценяване, носи 10 точки.
- Оценката от практическата част се изчислява по формулата $\frac{x}{10}$, където x е сумата от получените точки от задачите. Максималната оценка е Отличен 6.00.
- За успешно полагане на изпита са необходими поне 30 точки, като при събрани под 30 точки оценката е Слаб 2.00.

Изпитната комисия ви пожелава успешна работа!

Задача 1. *Решете задачата на езика C++. Решения на друг език носят нула точки. Отговорите на подточки 1А, 1Б, 1В и 1Г трябва да съвпадат с това, което би извела програмата. При несъответствие отговорът се оценява с нула точки. Решението на подточка 1Д трябва да бъде технически издържано (например не бива да изтича памет, трябва да се спазват добрите практики за структуриране на програмата и т.н.). Ако решението съдържа сериозни грешки, то се оценява с нула точки.*

1А) *(1 точка)* Какъв ще бъде изходът от изпълнението на следния фрагмент:

```
int calc(int a[5])
{
    int sum = 8;
    for (int i = 0; a[i]; ++i)
        sum += i;
    return sum;
}

int a[7] = {1, 2, 3};
std::cout << calc(a);
```

Отговор: _____

1Б) *(1 точка)* Какъв ще бъде изходът от изпълнението на следния програмен фрагмент?

```
char text[] = "hello", *p = text;
while (*p) std::cout << ++*p++;
```

Изберете един от следните отговори:

а) Грешка по време на компилация.

б) Грешка по време на изпълнение.

в) `ifmmp`

г) `el`

д) Безкраен цикъл.

1В) *(1 точка)* Какво ще изведе следният фрагмент (приемаме, че е част от валидна програма):

```
int *pt;
int a[3] = {4,  19,  13};
pt = &a[1];
pt += 1;
std::cout << *pt << std::endl;
```

Отговор: _____

1Г) *(1 точка)* В дадените по-долу празни места попълнете какви ще бъдат стойностите на елементите на двата масива A и B след обръщението към функцията f.

```
void f(int * arr1, const int * arr2)
{
    int *p1 = arr1;
    const int *p2 = arr2;

    while(*p2 >= 0)
    {
        *p1++ = *p2++;
    }
}

void main()
{
    int A[4] = {-1, -2, -3, -4};
    int B[4] = {10, 20, 30, -1};

    f(A, B);

}
```

Отговор:

`A[0] =` ____ `A[1] =` ____ `A[2] =` ____ `A[3] =` ____

`B[0] =` ____ `B[1] =` ____ `B[2] =` ____ `B[3] =` ____

1Д) *(6 точки)* Дадени са структура `Point`, описваща точка в декартова координатна система с координати `x` и `y` от тип `float`, и структура `Circle`, описваща окръжност с център `center` от тип `Point` и радиус `r` от тип `float`.

Да се дефинира функция `findRelativePosition`, която определя относителната позиция на две дадени окръжности една спрямо друга. Резултатът от изпълнението на функцията е стойност от изброения тип:

```
RelativePosition {NO_COMMON_POINTS, TOUCHING,
                  INTERSECTING, SAME}.
```

със следния смисъл:

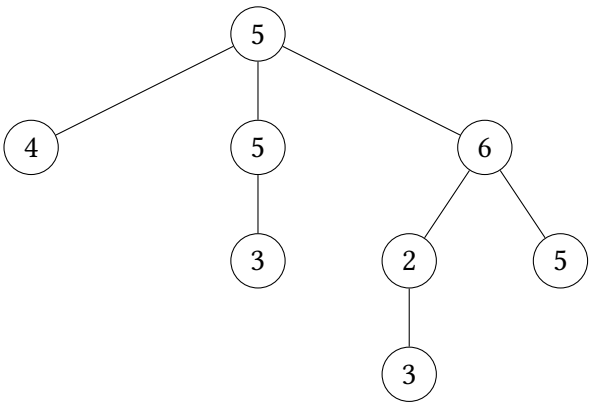
- `NO_COMMON_POINTS`: без общи точки
- `TOUCHING`: допиращи се
- `INTERSECTING`: пресичащи се
- `SAME`: съвпадат

Задача 2. Разглеждаме (кореново) дърво с върхове, чиито стойности са цели числа, като всеки връх може да има произволен брой деца. Такова дърво с $k + 1$ върха можем да представим като редица от двойки цели числа

$(level_0, value_0), ..., (level_k, value_k)$,

където за всяко $i : 0 \leq i \leq k$, $level_i$ показва нивото в дървото, на което се намира i-тият връх от представянето, а $value_i$ — неговата стойност. $(level_0, value_0)$ е коренът на дървото ($level_0 = 0$), а родителят на всеки връх $(level_i, value_i)$, $i > 0$, е най-близкият до него предходен елемент в редицата $(level_j, value_j)$, $0 \leq j < i$, такъв че $level_j + 1 = level_i$.

Пример: Редицата `(0,5),(1,4),(1,5),(2,3),(1,6),(2,2),(3,3),(2,5)` описва следното дърво:



Разбира се, не всяка такава редица е коректно представяне на дърво. Например, ако имаме два върха на ниво 0, това не е валидно представяне, тъй като коренът е единствен. Друг пример на некоректна редица е такава, в която за някой връх няма предхождащ го родител (например за върха със стойност 2 в : `(0,1),(2,2),(1,3)`).

А) (1 точка) Дефинирайте тип `TreeSeq` за представяне на редици от описания вид, който да ви е удобен за решаване на задачата. Можете да ползвате всички средтсва от стандартната библиотека.

Б) (1 точка) Дефинирайте подходящо представяне в паметта на връх на дърво от описания тип.

В) (7 точки) Напишете функция, която получава като аргумент редица от тип `TreeSeq`. Функцията да връща като резултат указател към корена на дърво, съответстващо на редицата, ако тя е валидно представяне на дърво. В противен случай да се връща `nullptr`.

Г) (1 точка) Демонстрирайте използването на тази функция в кратка програма.

Задачата да се реализира на езика C/C++.

Задача 3. *Задачата да се реши на езика Haskell. Отговорите си попълнете в посочените за тях полета. Оценява се само попълненото в полетата. Всичко друго писано по листа не носи точки.*

Под всеки от дадените по-долу изрази посочете каква ще бъде неговата оценка. За някои от изразите са дадени възможности, измежду които трябва да изберете. Други трябва да съобразите и да попълните сами. За изразите, които трябва да попълните, точки се дават само за напълно коректни отговори. Ако посоченият от вас отговор не съответства точно на това, което би извел Haskell интерпретаторът, той се оценява с нула точки.

А) *(1 точка)* Оградете оценката на израза:

```
foldr1 (&&) [False, False ..]
```

а) True
б) False
в) Изразът ще предизвика безкрайно изпълнение.

Б) *(1 точка)* Оградете оценката на израза:

```
foldr1 (&&) [True, True ..]
```

а) True
б) False
в) Изразът ще предизвика безкрайно изпълнение.

В) *(1 точка)* Попълнете оценката на израза:

```
filter (`elem` [10..20]) [1,5,10,100,20,15]
```

Отговор: _____

Г) *(1 точка)* Оградете оценката на израза:

```
negate $ max 10 20
```

а) 10
б) 20
в) −10
г) −20
д) Операторът $
е) Операторът $ не може да се използва инфиксно.

Д) *(1 точка)* Попълнете оценката на израза:

```
print (take 4 [1,3..])
```

Отговор: _____

Е) *(1 точка)* Попълнете оценката на израза::

```
(:[]) []
```

Отговор: _____

Ж) *(1 точка)* Попълнете оценката на израза:

```
map ($ 0) (map (+) [1..5])
```

Отговор: _____

З) *(1 точка)* Нека имаме следната функция:

```
f l = [ x+y | x <- l, y <- l]
```

Попълнете каква ще бъде оценката на израза:

```
f [1,10]
```

Отговор: _____

И) *(1 точка)* Нека имаме следната функция:

```
g ([]:_) = []
g x = (map head x) : g (map tail x)
```

Попълнете каква ще бъде оценката на израза:

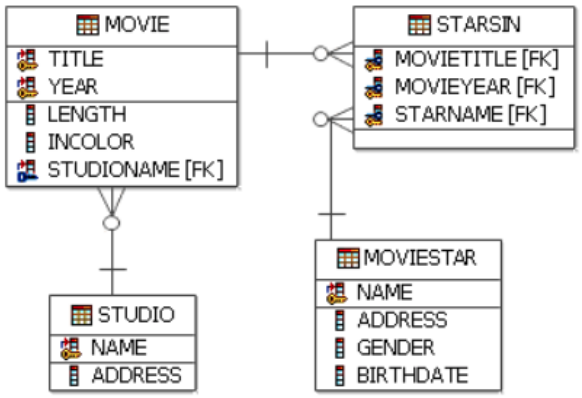
```
g [[1,2,3],[4,5,6]]
```

Отговор: _____

Й) *(1 точка)* В полето по-долу попълнете какъв е типът на функцията `g` от предишния въпрос. Опишете го така, че функцията да може да работи само и единствено за списъци от списъци, съдържащи елементи от класа `Num`.

Отговор: `g::` _____

Задача 4. Дадена е базата от данни **Movies**.



Таблицата **Studio** съдържа информация за филмови студиа:

- *name* — име, първичен ключ;
- *address* — адрес.

Таблицата **Movie** съдържа информация за филми. Колоните *title* и *year* заедно формират първичния ключ.

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

Таблицата **MovieStar** съдържа информация за филмови звезди:

- *name* — име, първичен ключ;
- *address* — адрес;
- *gender* — пол, 'F' за жена и 'M' за мъж;
- *birthdate* — рождена дата.

Таблицата **StarsIn** съдържа информация за участието на филмовите звезди във филмите. Трите колони заедно формират първичния ключ. Колоните *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.

Задача 4.1 (4 точки): Посочете заявката, която извежда имената на всички студиа, които имат поне един цветен филм и едновременно с това поне един филм с неизвестна дължина:

а)
```
SELECT studioName
  FROM Movie
  WHERE inColor != 'Y'
    AND studioName =
        (SELECT studioName
         FROM Movie
         WHERE length IS NULL);
```

б)
```
SELECT studioName
  FROM Movie
  WHERE inColor = 'Y'
    AND length = NULL;
```

в)
```
SELECT studioName
  FROM Movie
  WHERE inColor = 'Y'
  INTERSECT
  SELECT studioName
  FROM Movie
  WHERE length IS NULL;
```

г)
```
SELECT name
  FROM Studio
  LEFT JOIN Movie
    ON name = studioName
  WHERE inColor = 'Y'
    OR length = NULL;
```

Задача 4.2 (6 точки): Посочете заявката, която за всяка актриса извежда името и броя на черно-белите филми, в които е участвала. Ако за дадена актриса няма информация в какви филми е участвала или е играла само в цветни филми, срещу нейното име да се изведе числото 0.

а)
```
SELECT name, COUNT(DISTINCT title)
  FROM Movie
  RIGHT JOIN StarsIn
    ON title = movieTitle
      AND year = movieYear
  RIGHT JOIN MovieStar
    ON starName = name
  WHERE gender = 'F'
  GROUP BY name
  HAVING inColor = 'N';
```

б)
```
SELECT name, COUNT(title)
  FROM MovieStar
  LEFT JOIN StarsIn ON name = starName
  LEFT JOIN Movie
    ON movieTitle = title
        AND movieYear = year
        AND inColor = 'N'
 WHERE gender = 'F'
 GROUP BY name;
```

в)
```
SELECT name, COUNT(title)
  FROM MovieStar
  LEFT JOIN StarsIn ON name = starName
  LEFT JOIN Movie
    ON movieTitle = title
        AND movieYear = year
  HAVING inColor = 'N'
        AND gender = 'F'
  GROUP BY name;
```

г)
```
SELECT starName, COUNT(*)
  FROM StarsIn
  LEFT JOIN Movie
    ON title = movieTitle
        AND year = movieYear
  WHERE inColor = 'N' AND gender = 'F'
  GROUP BY name;
```

Задача 5. (10 точки) Даден е масив $A[1, 2, \dots, n]$ от цели числа и цяло число m. Предложете алгоритъм със сложност по време $O(n\sqrt{n})$, който връща

- 1, ако има два различни елемента на A със сума m,
- 0, ако няма такива елементи.

Не е необходимо да давате псевдокод, но описанието на алгоритъма трябва да е абсолютно ясно и недвусмислено. Накратко обосновете коректността му и сложността му по време.

Задача 6. (10 точки) Говорим за обикновени, неориентирани графи без примки. Нека $G = (V, E)$ е граф. *Антиклика* в G е всяко непразно $U \subseteq V$, такова че между никои два върха от U няма ребро. G се нарича *двуделен*, ако има разбиване $\{V_1, V_2\}$ на V, такова че за всяко ребро $e \in E$, единият край на e е във V_1, а другият край на e е във V_2.

Докажете, че G е двуделен тогава и само тогава, когато за всеки подграф H на G е вярно, че в G има антиклика, съдържаща поне половината от върховете на H.

Упътване: можете да ползвате наготово факта, че граф е двуделен тогава и само тогава, когато има нечетен цикъл.

Задача 7. В равнината е въведена декартова координатна система Oxy и е даден триъгълникът ABC. Известно е, че върхът A на триъгълника има координати $(-2, -2)$; медианата m_B на триъгълника, минаваща през върха B, има уравнение $x + 9y + 8 = 0$, а ъглополовящата l_B на вътрешният ъгъл на триъгълника при върха B има уравнение $y + 1 = 0$.

a) (8 точки) Да се намерят координатите на върховете B и C.

b) (2 точки) Да се намери лицето на триъгълника ABC.

Чернова