СОФИЙСКИ УНИВЕРСИТЕТ
“СВ. КЛИМЕНТ ОХРИДСКИ”

ФАКУЛТЕТ ПО МАТЕМАТИКА
И ИНФОРМАТИКА

ДЪРЖАВЕН ИЗПИТ
ЗА ПОЛУЧАВАНЕ НА ОКС “БАКАЛАВЪР ПО ИНФОРМАТИКА”
22-23.03.2008 г.

ЧАСТ I (ПРАКТИЧЕСКИ ЗАДАЧИ)

Време за работа – 3 часа

Драги абсолвенти:
- *Попълнете факултетния си номер на всички страници;*
- *За всяка от задачите,ползвайте за чернова и за решение само листите, на които е изписано условието на съответната задача.*

Изпитната комисия ви пожелава успешна работа

Задача 1. (5 т.) Ако, в резултат на успешна компилация на зададения по-долу код на C, е създаден изпълним файл с име **progA** и в текущата директория има текстов файл **fileA.txt** със съдържание

```
xyz123-y
111bbb
Zzzzzz5yyyyy
```

напишете вдясно какво ще се изведе на стандартния изход в случай на успешно изпълнение на **progA** след стартиране с командния ред

./progA fileA.txt

```
#include      <stdio.h>
#include      <fcntl.h>

main( int argc, char *argv[] )
{
      int fd,  i = 0 , j ;
      char c;

      if  (  (  fd = open ( argv[1], O_RDONLY )  )  == -1 )
      {  printf (“\n Cannot open %s  ”, argv[1] );    exit(1);          }

      while  (   read ( fd, &c, 1 )    )
      {       ++i;
              if  (  c = = '\n' )
              {    if ( i < 10 )
                      for (j =  i ; j < 15 ; j++ )
                        write(1,”$”,1);
                   i=0;
               }
              write(1,&c,1);
      }
      close(fd);
}
```

Задача 2. (5 т.) Да се провери пълно ли е множеството от двоични функции A:

а) $A = (L \circ M) \circ (S \setminus T_0)$

б) $A = (M \circ S) \circ (T_0 \setminus M) \circ (T_1 \circ S)$

Задача 3. (4 т.) Даден е крайният детерминиран автомат A:

$$A = < \{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8\}, \{x, y, z\}, q_0, \delta, \{q_0, q_3, q_4, q_6, q_7, \} >$$

с функция на преходите δ, зададена чрез таблицата:

	x	y	z
q_0	q_1	q_1	q_4
q_1	q_0	q_2	q_2
q_2	q_3	q_1	q_1
q_3	q_1	q_1	q_4
q_4	q_5	q_2	q_3
q_5	q_7	q_8	q_5
q_6	q_5	q_1	q_7
q_7	q_2	q_2	q_6
q_8	q_6	q_8	q_6

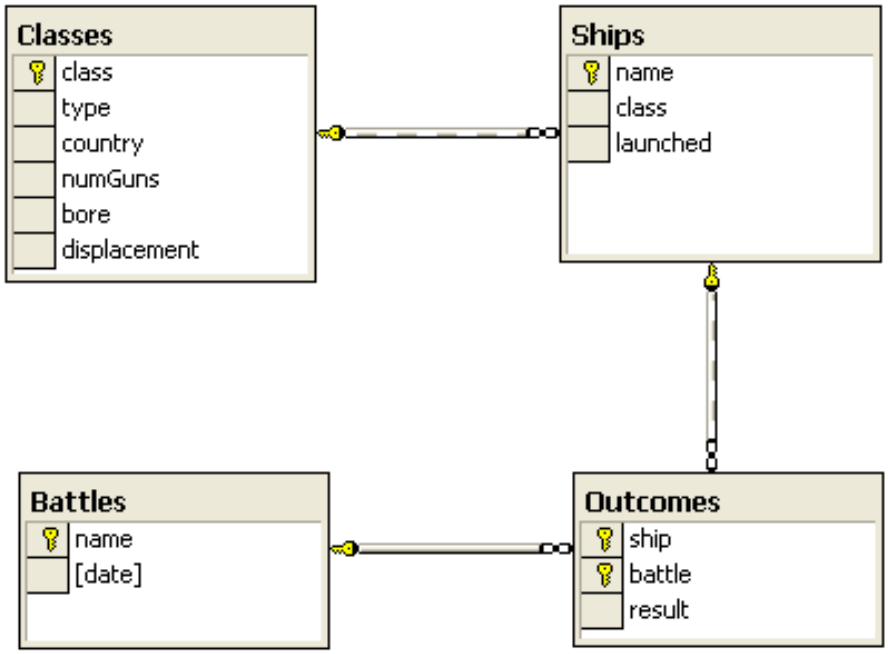
Да се построи минимален детерминиран краен автомат, еквивалентен на дадения.

ЧЕРНОВА ЗА ЗАДАЧИ 2 и 3

Задача 4. (6 т.)
В базата данни със схема:
Classes (class, type, country, numGuns, bore, displacement)
Ships (name, class, launched)
Battles (name, date)
Outcomes (ship, battle, result)

се съхранява информация за кораби и тяхното участие в битки по време на Втората Световна Война.
Всеки кораб е построен по определен стереотип, определящ класа на кораба. Обикновено класът носи името на първия построен кораб от този клас.

- Таблицата **Classes** съдържа информация за име на класа, тип (‘**bb**’ за бойни кораби и ‘**bc**’ за бойни крайцери), страната, която строи такива кораби, броя на основните оръдия, калибъра им (диаметъра на отвора на оръдието в инчове) и водоизместимостта (тегло в тонове).
- Таблицата **Ships** съдържа информация за име на кораб, име на неговия клас и годината, в която корабът е пуснат на вода.
- Таблицата **Battles** съхранява имена и дати на провеждане на битки.
- Таблицата **Outcomes** съдържа информация за резултатата от участието на даден кораб в дадена битка (потънал - ‘**sunk**’, повреден – ‘**damaged**’, победил – ‘**ok**’).



(1) (1 т) Посочете заявката, която извежда имената на всички кораби, в чието име се среща низа **king** (независимо с малки или големи букви)

a)
```
SELECT ss.name
FROM ships ss
WHERE LOWER(ss.name) = '%king%';
```

b)
```
SELECT ss.name
FROM ships ss
WHERE ss.name LIKE 'king%';
SELECT ss.name
FROM ships ss
WHERE LOWER(ss.name) LIKE '%king%';
```

c)
```
SELECT ss.name
FROM ships ss
WHERE ss.name LIKE '*king*';
```

(2) (3т) Посочете заявката, която извежда имената на класовете, за които няма кораб, пуснат на вода след 1921 г. Ако за класа няма пуснат никакъв кораб, той също трябва да излезе в резултата.

a)
```
SELECT c.class
FROM classes c
WHERE NOT EXISTS
  ( SELECT 1
    FROM ships t
    WHERE t.class=c.class AND t.launched > 1921);
```

b)
```
SELECT c.class
FROM classes c
WHERE ALL (SELECT t.name
           FROM ships t
           WHERE t.class=c.class AND t.launched <= 1921);
```

c)
```
SELECT t.class
FROM ships t
GROUP BY t.class
HAVING MAX(t.launched)<=1921;
```

d)
```
SELECT c.class
FROM classes c
LEFT JOIN ships t ON t.class = c.class
WHERE t.launched > 1921;
```

(3) (2т) Посочете заявката, която извежда броя на потъналите японски кораби за всяка проведена битка с поне един потънал японски кораб:

a)
```
SELECT battle, COUNT(s.name)
FROM classes c
JOIN ships s ON s.class = c.class
JOIN outcomes o ON s.name = o.ship
WHERE c.country = 'Japan' AND o.result = 'sunk';
```

b)
```
SELECT battle, COUNT(*)
FROM classes c
JOIN ships s ON s.class = c.class
JOIN outcomes o ON s.name = o.ship
WHERE c.country = 'Japan' AND o.result = 'sunk'
GROUP BY battle;
```

c)
```
SELECT battle, COUNT(s.name)
FROM classes c
JOIN ships s ON s.class = c.class
JOIN outcomes o ON s.name = o.ship
WHERE c.country = 'Japan'
GROUP BY battle
HAVING o.result = 'sunk';
```

d)
```
SELECT battle, COUNT(s.name)
FROM outcomes o
LEFT JOIN ships s ON s.name = o.ship AND o.result ='sunk'
LEFT JOIN classes c ON s.class = c.class AND
                       c.country = 'Japan'
GROUP BY battle;
```

Задача 5. (4 т.) Нека **p** е предикат, дефиниран със следните клаузи на Пролог:

```
p( X, L, [ X| L]).
p( X, [ Y| L], [ Y| L1]) :- p( X, L, L1).
```

1) **(2 т.)** Кой от изброените списъци **не може** да е отговор на целта

```
?- p( a, [ b, a, b],  L).
```

а) L = [a, b, a, b]
б) L = [b, a, b, a]
в) L = [a, b, b, a]
г) L = [b, a, a, b]

2) **(2 т.)** Кой от изброените списъци **е** отговор на целта

```
?- p( a, L, [ b, a, b] ).
```

а) L = [a, b, a, b]
б) L = [b, a, b]
в) L = [a, b]
г) L = [b, b]

Задача 6. (2 т.) Коя от изброените формули е тавтология?

а) $\forall X \exists Y\ p(X, Y) \Rightarrow \exists Y \forall X\ p(X, Y)$
б) $\forall X \exists Y\ p(X, Y) \Leftrightarrow \exists Y \forall X\ p(X, Y)$
в) $\exists Y \forall X\ p(X, Y) \Rightarrow \forall X \exists Y\ p(X, Y)$
г) $\exists Y \forall X\ p(X, Y) \Leftrightarrow \forall X \exists Y\ p(X, Y)$

Задача 7. (5 т.) Нека Γ е оператор, преработващ частични функции над целите числа, който се дефинира по следния начин:

$\Gamma(f)(x) \cong$ if $x > 100$ then $x - 10$
else $f(f(x + 11))$.

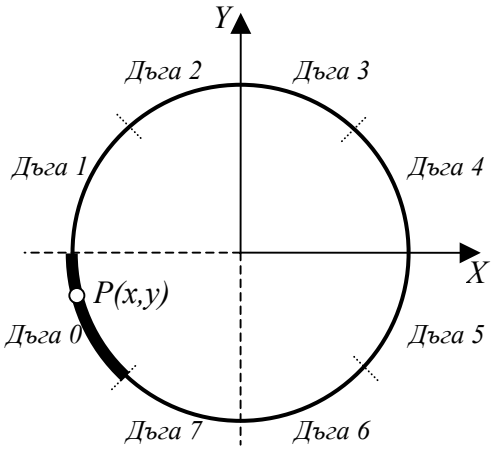
Докажете, че за най-малката му неподвижна точка f_Γ е изпълнено условието:

$$\forall x \in Z(x \leq 100 \ \& \ !f_\Gamma(x) \ \Rightarrow \ f_\Gamma(x) = 91).$$

Задача 8. (4 т.)

Нека имаме процедурата **drawCircle** за рисуване на окръжност с център (0,0). За да се спести от изчисляването на координатите на рисуваните пиксели, тя пресмята координатите само на 45 точки от дъга 0, която е 1/8 от цялата окръжност. За всяка пресметната точка **P(x,y)** от дъга 0 процедурата рисува осем пиксела – по един от всяка дъга.

В дефиницията на тази процедура командата **plot(x,y)** рисува точка **P(x,y)** от дъга 0, а **plot(-x,-y)** рисува съответната точка от дъга 4. Определете и попълнете в таблицата коя дъга се рисува от всяка от останалите шест команди **plot.**




```
void drawCircle;
{ int i;
  int x;
  int y;

  for( i=0; i<45; i++ )
  {
    X = <изчисляване на X координатата>;
    y = <изчисляване на Y координатата>;
    plot(x,y);
    plot(y,x);
    plot(-x,y);
    plot(-y,x);
    plot(x,-y);
    plot(y,-x);
    plot(-x,-y);
    plot(-y,-x);
  }
}
```

Команда:	Рисува дъга:
plot(x,y)	**0**
plot(y,x)	
plot(-x,y)	
plot(-y,x)	
plot(x,-y)	
plot(y,-x)	
plot(-x,-y)	**4**
plot(-y,-x)	

Задача 9. (3 т.) Като използвате единствено процедурите cons, car и cdr (и техните производни) чрез обръщения към l напишете израз, който:

1. конструира дадения списък, където l=(one two three):

а) (one (two three))

……

……

б) (one (two) three)

……

……

в) ((one two three))

……

……

2. има оценка стойността Harry, ако l има вида:

а) (Ann and Harry)

……

……

б) (((Ann)) and ((Harry)))

……

……

Задача 10. (4 т.) Оценете изразите:

```
((lambda (x y) (x y 8))(lambda (x y) (/ x y)) 2)
```

……

```
(accumulate + 1 (filter even? '(1 44 73 12 16 7)))
```

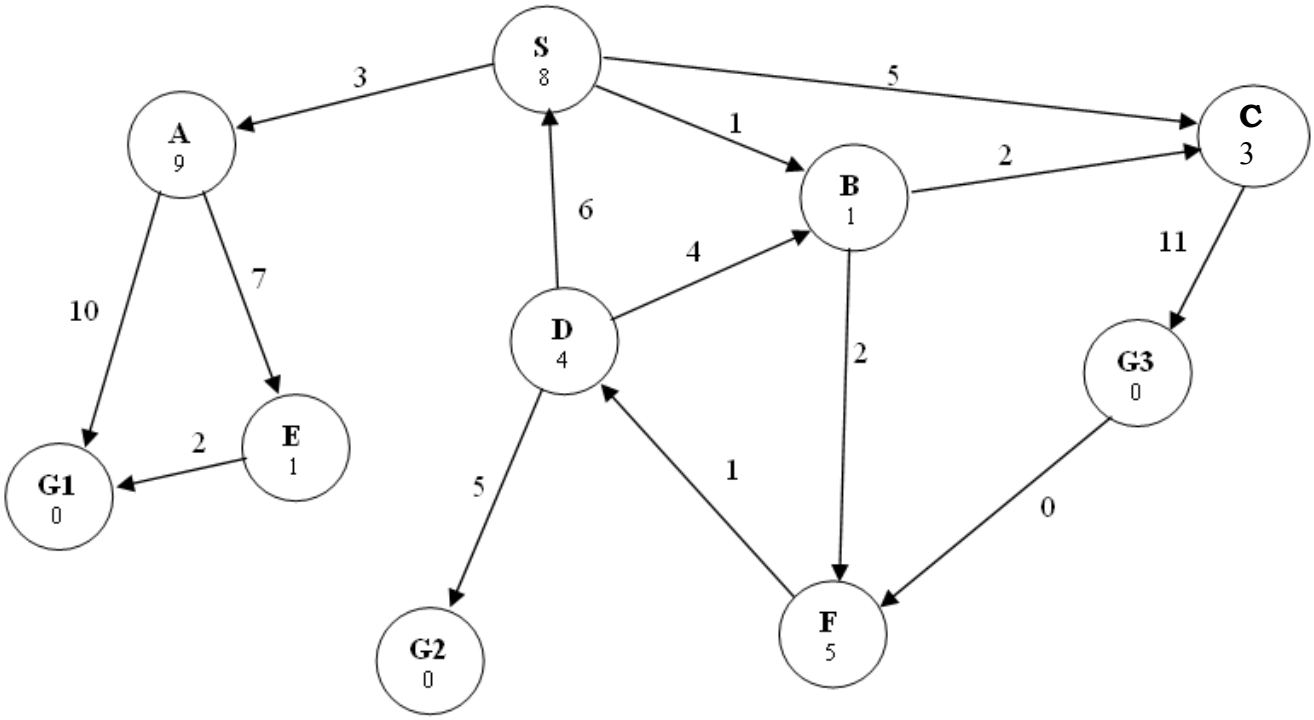
……

```
(let* ((x (list (lambda (x) (x 3 4))))
       (x (list (lambda (x) (+ x 5)) ((car x) +))))
       (cadr x))
```

……

Задача 11. (8 т.) Дадено е пространство от състояния представено чрез ориентиран граф (фиг. 2), където S е начално състояние, а G1, G2 и G3 са целеви състояния. Дъгите са насочени и за всяка от тях е зададено теглото й. За всяко от състоянията (вътре във възела заедно с името) е зададена евристична оценка до целевите състояния.

	Коя цел е достигната	**Кои състояния са обходени в процеса на търсенето**
Iterative Deepening		
Breadth-First Search		
Hill Climbing		
*A**		



фиг. 2

Задача 12. (5 т.) Даден е следният програмен фрагмент, написан на езика C++

```
int A[3][3]={1,2,3,4,5,6,7,8,9}, B[3][3] ;
int S, i, j ;
S = i = j = 0 ;
for (   ; i <2 ;   )
     for (   ; j<2 ;  )
          S += A[i++][j++] ;
cout ” S= ” << S ;
```

a) Какъв ще бъде резултатът от изпълнението на фрагмента?

b) Предложете нов вариант на този фрагмент, който за матрица B (предполага се, че нейните елементи вече са въведени) намира и извежда сумата на всички елементи, до срещането на нулев елемент, ако в матрицата има такъв.

Задача 13. (5 т.) Дадена е свързана опашка, елементите на която се описват със структурата:

```
struct QElem
{
    int     data
    QElem * next;
};
```

Самата опашка е представена посредством указатели към първия и последния си елемент:

```
struct Queue
{
    QElem * front;
    QElem * rear;
};
```

Дадена е и следната функция за инициализиране

```
void initialize (Queue* queue)
{
    queue->front = 0;
    queue->rear = 0;
}
```

а) (2 т.) Да се напише функцията

```
void enqueue(Queue* queue, int data);
```

която добавя елемента `data` в опашката.

б) (2 т.) Да се напише функцията

```
int dequeue(Queue* queue, int* data);
```

която премахва елемент от опашката и записва стойността му в `data`. Връща 0 ако не е успяла да премахне елемента и 1 ако е успяла.

в) (1 т.) Да се напише функцията

```
void uninitialize(Queue* queue);
```

която премахва всички елементи на опашката.

Задача 14. (3 т.) Даден е едносвързан списък, елементите на който се описват със структурата:

```
struct ListElem
{
    double     data;
    ListElem * next;
};
```

Да се напише функция

```
int insert(ListElem * start, double key, double data);
```

която вмъква в списъка с начало сочено от `start` елемент с данни `data` след всяко срещане на елемент с данни `key`. Връща броя на вмъкнатите елементи.

Задача 15. (6 т.)Задача Дадено е двоично наредено дърво, елементите на което се описват със структурата:

```
struct TreeNode
{
    int        data;
    TreeNode * left;
    TreeNode * right;
};
```

За всеки елемент в лявото му поддърво се съдържат елементи по-малки или равни на този елемент, а в дясното - по-големи или равни.

а) (3 т.) Да се напише функция

```
double processEven(const TreeNode * root);
```

която извежда в нарастващ ред всички четни елементи на дървото с корен сочен от `root` и като резултат връща средно аритметичното им.

б) (3 т.) Да се напише функция

```
void print(const TreeNode * root, int k);
```

която извежда на екрана всички елементи на дървото с корен сочен от `root`, които са на дълбочина `k`. Приемаме, че коренът има дълбочина 0.

Задача 16. (13 т.) Дадена е следната програма на C++, която е компилирана с MS Visual Studio 6.0. Моля попълнете в съответните полета какво се отпечатва на екрана като резултат от изпълнението на съответните конструкции.

В полетата, обозначени със "свързване", попълнете вида на свързването на съответния метод ("С" – статично, "Д" – динамично).

```
#include <iostream.h>
class Base {
public:
      Base () {cout << "Base\n";}
      virtual void f ()
      {
            cout << "Base::f()\n";
            g();
            h();
      }
      virtual void g () {cout << "Base::g()\n";}
      void h () {cout << "Base::h()\n";}
      ~Base () {cout << "Base::~Base()\n";}
};
class Der1 : public Base {
public:
      Der1 () {cout << "Der1\n";}
      void g () {cout << "Der1::g()\n";}
      void h () {cout << "Der1::h()\n";}
      virtual ~Der1 () {cout << "Der1::~Der1()\n";}
};
class Der2 : public Der1 {
public:
      void f ()
      {
            cout << "Der2::f()\n";
            g();
            h();
      }
      void h () {cout << "Der2::h()\n";}
      ~Der2 () {cout << "Der2::~Der2()\n";}
};
void main ()
{
     Der1 d1;

     Der2 d2;

     Base *bp = new Der2;

     Der1 *d1p = new Der2;

     bp->f ();
```

Свързване:	а)
Свързване:	б)

Свързване:	в)
Свързване:	г)

д)
е)
ж)
з)
Свързване: и)
й)

```
      bp->h ();

      d1p->f ();

      d1p->h ();

      d1.f();

      d1.h();

      d2.f ();

      d2.h ();

      delete bp;

      delete d1p;

}
```

Свързване: к)
л)

Свързване: м)
н)

Свързване: о)
п)

р)

с)

т)

у)

ф)

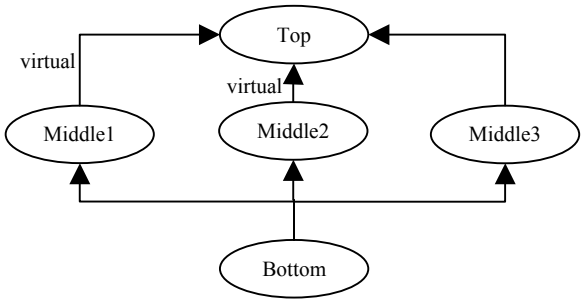
х)

ц)

Задача 17. (9 т.) Дадена следната програма, компилирана с Visual Studio 6.0, реализираща йерархията на фиг. 3. Какъв е изходът от програмата?

```
#include <iostream.h>
class Top {
public:
      Top (){cout << "Top::Top()\n";}
      Top (int x){cout << "Top::Top("<<x<<")\n";}
};
class Middle1: virtual public Top {
public:
      Middle1():Top(1){cout << "Middle1::Middle1()\n";}
};
class Middle2: virtual public Top {
public:
      Middle2():Top(2){cout << "Middle2::Middle2()\n";}
};
class Middle3: public Top {
public:
      Middle3():Top(3){cout << "Middle3::Middle3()\n";}
};
class Bottom : public Middle1,
                    public Middle2,
                    public Middle3 {
public:
      Bottom (){cout<<"Bottom::~Bottom()";}
};

void main (){
      Bottom b;
}
```



Фиг. 3

Задача 18. (9 т.) Дадена е следната програма, компилирана с Microsoft Visual Studio 6.0. Попълнете празните полета с изхода, предизвикан от съответните програмни конструкции.

```
#include <iostream.h>
class A {
public:
      A () {cout << "A::A()\n";}
      A (const A&) {cout << "A::A(const A&)\n";}
      A (int) {cout << "A::A(int)\n";}
      operator int () {cout<<"A::operator int()\n"; return 0;}
      ~A () {cout << "A::~A()\n";}
};

void f (A){cout << "f(A)\n";}
void g (const A&){cout << "f(const A&)\n";}
void h (int){cout << "f(int)\n";}
A i (){cout << "i()\n"; return A();}


void main (){

      A a;
```

а)

```
      f(a);
```

б)

```
      f(0);
```

в)

```
      g(a);
```

г)

```
      h(a);
```

д)

```
      a = i();
```

е)

```
}
```

ж)