

Task1 (12):				Task4 (12):			
Task2 (12):				Task5 (16):			
Task3 (12):				Task6 (18):			
Task7 (18)							
Task1	Task2	Task3	Task4	Task5	Task6	Task7	Sum()

Task1. What is the output for the following program?

```
#include <iostream>

using namespace std;

class Kop{
public:
    int x;
    Kop(){x = 3;}
    virtual int a();
    int b();
    void operator() (Kop& x);
    bool operator|| (Kop& rh) {
        return (x-- >= (rh.x)++) || (--x > (rh.x)++);
    }
};

int Kop::a(){cout << this->x << "KopA\n"; return b();}
int Kop::b(){cout << x << "KopB\n"; return this->x;}
void Kop::operator() (Kop& x){
    this->x = x.a() * x.b();
    cout << this->x << endl;
}

class Jada: public Kop{
public:
    int a();
    int b();
};

int Jada::a(){    cout << "JadaA\n";return b()*100;}
int Jada::b(){    cout << "JadaB\n";return this->x * 10;}

int main(){
    Kop obj;
    Jada obj1;
    Kop *d = new Jada();
    obj.a(); d->a();  obj.b();
    if (obj || obj1)
        obj.x = (obj.x)++*--(obj1.x);
    d->b(); obj(*d); (*d)(obj);
    return 0;
}
```

Task2. What is the output for the following program?

```
#include <iostream>

using namespace std;

#define jada "Kopaonik"
#define ComplexADD(a,b,c) a+b*c

static int c=2;
static int b;

class X {
    int *pi;
public:
    int x;
    int y;
    int z;
    X():pi(new int(3)), x(c++), y(c++), z(ComplexADD(2,3,++c+1)) {b=0;};
    X(int i) : pi(new int(i)), x(c++), z(c++), y(ComplexADD(3,2,++c+1)) {b++;}
    X(const X &ob) : pi(new int(*ob.pi)) {
        cout << "kop" << endl; b*=2; x=ComplexADD(c, y+z, z+b);
    }
    X& operator= (const X& x);
    ~X() {delete pi;}
};

X& X::operator= (const X& x) {
    if (this != &x) {
        delete pi;
        pi = new int(*x.pi);
        cout << "ELE" << endl;
    };
    return *this;
}

X funF(X x){X Xnew(5); x=Xnew; return x;}

void g() {
    X xa=3, xb=5;
    X xc = xa;
    xa = funF(xb);
    xc = xa;
}

int main(int argc, char* argv[]){
    X d;
    g();
    cout << b << c << endl;
    cout << d.x << d.y << d.z << endl;
    return 0;
}
```

Task3. What is the output for the following program?

```
#include <iostream>
using namespace std;

int data = 0;

class A
{
    int x;
    virtual int f( int );
public:
    A(){data++;}
    A( int a ) : x(a) {}
    double f( double );

    A& operator++( ) {cout << "Kop1 " << data-- << endl; A* b= new A; return *b;}
    A operator++(int) {cout << "Kop2 " << data++ << endl; A* b= new A; return *b;}
    A& operator--( ) {cout << "Kop3 " << ++data << endl; A* b= new A; return *b;}
    A operator--(int) {cout << "Kop4 " << --data << endl; A* b= new A; return *b;}
    void operator-(A& b) { data++;}
    ~A() {data++;}
};

int A::f( int a ){ return a+x;}
double A::f( double a ){ return a*x + f(5);}

class B
{
public:
    virtual void g(int d = 2) = 0;
};

class C : public A, public B
{
public:
    int x;
    int f( int );
public:
    C():x(data){};
    C ( int a, int b ) : A(a), x(b) {};
    double f( double );
    void g(int d = 2) { A a; a++; data+=d--;;}
};

int C::f( int a = 5){ A b; b++; data+=a--; return a-x;}
double C::f( double a ){ return a/x+f(5); f(1);}

void main()
{
    A i, j;
    B *c = new C;
    A x(2);
    C y(2,4);
    A* pb = new C(2,4);

    cout << x.f( 2.4 ) << endl;
    cout << y.f( 2.4 ) << endl;
    cout << pb->f( 1.2 ) << endl;
    --++i----++j--;
    y.f(data); y.g(data);
    cout << data;
}
```

Task4. What is the output for the following program?

```
#include <iostream>

using namespace std;

int id = 0;

class CrAde {
public:
    CrAde():objectID(id) { id = 1;}
    CrAde(int objectNumber) {
        objectID = objectNumber;
        cout << objectID << " conC&D" << endl;
    }

    ~CrAde() {cout << objectID << " desC&D" << endl;}

private:
    int objectID;
};

void create( void );

CrAde a(id++);
static CrAde b(id++);

class CD1 : virtual public CrAde {
public:
    CD1( int i): CrAde(i) {CrAde b(i);}
    virtual ~CD1() {CrAde b(id++); }
};

class CD2 : virtual public CrAde {
    CrAde a;

public:
    CD2(int i) : CrAde(i) {}
    ~CD2() { cout << id << " desC&D" << endl; CrAde b(id++); }
};

class CD3 : public CD1, virtual public CrAde, public CD2 {
public:
    CD3(int i) : CD1(i), CD2(i), CrAde(i) { static CrAde a(id++);}
    ~CD3() {cout << id << " desC&D" << endl; }

};

void main( ) {
    CrAde b(id++);
    CD3 a(id++);
    static CrAde c(id++);
    create();
}

void create( void ) {
    static CrAde b(id++);
    CrAde c(id++);
}
```

Task5. What is the output for the following program?

```
#include <iostream>
using namespace std;

class Operate {
public:
    virtual void execute(int data[], int n) = 0;
};

class Add : public Operate {
    int *coef;
    int n;
public:
    Add() { n = 0; coef = NULL; };
    Add(int *d, int no) {
        n = no;
        coef = new int[no];
        for (int i=0; i<n; i++)
            coef[i] = d[i];
    };
    void execute(int data[], int n) {
        for (int i=0; i<n; i++)
            data[i] += coef[i];
    };
};

class Square : public Operate {
    Operate *step;
    virtual void process(int data[], int n, int coef[]) {
        for (int i=0; i<n; i++)
            data[i] += coef[i];
    };
public:
    Square() { step = NULL; };
    Square(Operate *st) { step = st; };
    Operate* set(Operate *st) { step = st; return this; };
    Operate* get() { return step; };
    void execute(int data[], int n) {
        int *d = new int[n];
        for (int i=0; i<n; i++)
            d[i] = 0;
        step->execute(d, n);
        process(data, n, d);
        delete[] d;
    };
};

class Exponent : public Square {
    int axis;
    void process(int data[], int n, int coef[]) {
        for (int i=0; i<n; i++)
            data[i] += (i==axis ? -1 : 1) * coef[i];
    };
public:
    Exponent(Operate *st, int ax=0) : Square(st) { axis = ax; };
};

int limit[2] = {9, 14};
int mat[9][14];

int check(int data[], int d[], int n) {
    bool eq = true;
    for (int i=0; i<n; i++, eq=eq&&data[i]==d[i])
        if (d[i] < 0 || d[i] >= limit[(i+1)%2]) return i+2;
    return eq ? 1 : mat[d[1]][d[0]];
}
```

```

void main() {
    int n = 2, data[2] = {}, d[2] = {}, test[2] = {12, 8}, m = 4;
    int a[] = {2, 1};
    int b[] = {1, 2};
    Operate *oper[2];
    oper[0] = new Add(a, n);
    oper[1] = new Add(b, n);
    Square *square[2];
    square[0] = new Exponent(oper[0]);
    square[1] = new Exponent(oper[0], 1);

    int vn[] = {0,0,0,1,1,0,0,0,0,1,1,0,0,1};
    int vp[] = {0,0,0,1,1,1,0,0,0,1,0,0,0,0};
    for (int i=2; i<limit[0]-1; i++)
        for (int j=0; j<limit[1]; j++)
            if (i % 2) mat[i][j] = vn[j];
            else mat[i][j] = vp[j];

    bool end = false;
    int ind = 0, ret = 0;
    Square *sqrt = new Square(oper[0]);
    Square *ex = new Exponent(oper[0], 1);
    while (!end) {
        for (int i=0, ret=1; ret && i<n*n*n; i++) {
            for (int j=0; j<n; j++)
                d[j] = data[j];
            if (i/4)
                if ((i/2)&1 ^ (i/4)&1)
                    sqrt->set(square[(i+1)%2]->set(
                        square[i%2]->set(oper[i&1^(i/2)&1])));
                else
                    sqrt->set(square[i/4]->set(oper[i&1^(i/2)&1]));
            else
                if ((i/2)&1 ^ (i/4)&1)
                    sqrt->set(square[0]->set(oper[i&1^(i/2)&1]));
                else
                    sqrt->set(oper[i&1^(i/2)&1]);
            if (ind % 2)
                sqrt->set(ex->set(sqrt->get()));
            sqrt->execute(d, n);
            if ((ret = check(data, d, n)) == 3) {ind++; i=-1;};
        }
        end = true;
        for (int i=0; i<n; i++) {
            cout << (data[i] = d[i]) << " ";
            end = end && test[i] == data[i];
        }
        cout << endl;
    }

    delete sqrt;
    delete ex;
    for (int i=0; i<n; i++)
        delete oper[i];
}

```

Task6. What is the output for the following program?

```
#include <iostream>
using namespace std;

template <class T> class ENode {
public:
    T info;
    ENode<T>* next;
    ENode<T>* prev;
    ENode(T i) { info = i; next = NULL; prev = NULL; };
    ENode(ENode<T>* nx, ENode<T>* pr) { next = nx; prev = pr; };
};

template <class T> ostream& operator << (ostream& output, const ENode<T>& node)
{ output << node.info; return output; };

template <class T> class EList {
protected:
    ENode<T> *head, *tail;
public:
    EList() { head = tail = NULL; }
    bool isEmpty() { return head == NULL; }
    T getHead() {
        if (head != NULL) return head->info;
        else return T();
    }
    void insertHead(ENode<T> *ptr, bool b=true) {
        par = b;
        ENode<T> *ptr1 = head, *prev1 = NULL;
        while (ptr1!=NULL && ptr->info>ptr1->info) {
            prev1 = ptr1; ptr1 = ptr1->next;
        }
        if (prev1 != NULL) {
            prev1->next = ptr; ptr->next = ptr1;
        } else {
            head = ptr; ptr->next = ptr1;
        }
    }
    ENode<T>* deleteHead() {
        ENode<T>* tmp = head;
        if (head == tail) head = tail = NULL;
        else head = head->next;
        return tmp;
    }
    void traverse(ENode<T> *pNode) {
        int cnt = 0;
        for (ENode<T> *ptr=head; ptr!=NULL; ptr=ptr->next, cnt++) {
            if (pNode->info <= ptr->info) {
                ENode<T> *pN1, *pN2;
                if (pNode->info < ptr->info) {
                    pN1 = pNode; pN2 = ptr;
                } else {
                    pN1 = ptr; pN2 = pNode;
                }
                ENode<T> *pE = pN1->prev, *pEp = NULL;
                while (pE != NULL && pE->next->info < pN2->info) {
                    pEp = pE; pE = pE->prev;
                }
                if (pEp == NULL)
                    pN1->prev = new ENode<T>(pN2, pN1->prev);
                else
                    pEp->prev = new ENode<T>(pN2, pE);
            }
        }
        cout << (cnt/4&0x1) << (cnt/2&0x1) << (cnt&0x1) << endl;
    }
};
```



```

void print() {
    for (ENode<T>* tmp = head; tmp != NULL; tmp = tmp->next) {
        cout << endl << *tmp << " ";
        for (ENode<T>* ptr = tmp->prev; ptr != NULL; ptr = ptr->prev)
            cout << *(ptr->next) << " ";
    }
};

bool par = false;

class EMap {
public:
    int k1, v1, k2, v2;

    EMap() { k1 = v1 = k2 = v2 = 0; };
    EMap(int x1, int y1, int x2, int y2)
        { k1 = x1; v1 = y1; k2 = x2; v2 = y2; };
    EMap(const EMap& rc)
        { k1 = rc.k1; v1 = rc.v1; k2 = rc.k2; v2 = rc.v2; };
    bool operator < (const EMap& rc) { return k1 < rc.k1; };
    bool operator > (const EMap& rc) { return par ? v1 > rc.v1 : v2 > rc.v2; };
    bool operator <= (const EMap& rc) { return k1 <= rc.k2 && k2 >= rc.k1; };
};

ostream& operator << (ostream& output, const EMap& rc)
{ output << " ( " << rc.k1 << " " << rc.v1 << " " << rc.k2 << " "
  << rc.v2 << " ) " ; return output; };

void createList(ENode<EMap> *nodes[], int n) {
    EList<EMap> ltmp, llist;
    for (int i=0; i<n; i++) {
        bool bStop = false;
        while (!bStop && !ltmp.isEmpty()) {
            if (nodes[i]->info.v1 > ltmp.getHead().v2) {
                ENode<EMap> *pNode = ltmp.deleteHead();
                llist.insertHead(pNode);
            } else {
                bStop = true;
            }
        }
        ENode<EMap> *pNode = nodes[i];
        ltmp.traverse(pNode);
        ltmp.insertHead(pNode, false);
    }
    while (!ltmp.isEmpty())
        llist.insertHead(ltmp.deleteHead());
    llist.print();
}

void main() {
    ENode<EMap> *nodes[8];
    nodes[0] = new ENode<EMap>(EMap(10, 10, 20, 20));
    nodes[1] = new ENode<EMap>(EMap(15, 15, 45, 25));
    nodes[2] = new ENode<EMap>(EMap(25, 30, 35, 45));
    nodes[3] = new ENode<EMap>(EMap(15, 35, 25, 50));
    nodes[4] = new ENode<EMap>(EMap(20, 40, 30, 55));
    nodes[5] = new ENode<EMap>(EMap(40, 35, 50, 70));
    nodes[6] = new ENode<EMap>(EMap(0, 60, 25, 80));
    nodes[7] = new ENode<EMap>(EMap(15, 70, 20, 75));
    createList(nodes, 8);
}

```

Task7. What is the output for the following program?

```
#include <iostream>
using namespace std;

template <class T>
class EVector {
    int n, sz;
    T *ar;
public:
    EVector<T>(int no) { ar = new T[++no]; n=no; sz=0; };
    ~EVector<T>() { delete[] ar; };
    EVector<T>* clear() { for(int i=0; i<n; i++) ar[i]=T(); return this; };
    void insert(T el) { ar[++sz] = el; };
    bool operator < (const EVector<T>& bTree) { return ar[1] < bTree.ar[1]; };
    bool operator > (const EVector<T>& bTree) { return ar[1] > bTree.ar[1]; };
    void copy(EVector<T>* vec, bool mode) {
        int level = 0;
        int start = 1;
        int tmp = mode ? 3 : 2;
        while (level < vec->sz) {
            for (int part=start; part<start<<1; part++) {
                ar[tmp+part-start] = vec->ar[part];
            }
            level++;
            start <= 1;
            tmp *= 2;
        }
        sz = max(sz, vec->sz + 1);
        if (mode) ar[1] = ar[2] + ar[3];
    };
    int inverse(T& val, int tmp, int& res) {
        int ret = 0;
        if (tmp > n || ar[tmp] == T()) { return 0; };
        if (ar[tmp] == val) { return 1; };
        if (ret += inverse(val, 2*tmp, res)) {
            res <= 2; res = res | 0x1; return ++ret; }
        if (ret += inverse(val, 2*tmp+1, res)) {
            res <= 2; res = res | 0x2; return ++ret; }
        return false;
    }
};

template <class T>
class EMatrix {
    int n, sz;
    EVector<T> **ar;
public:
    EMatrix(int no) { ar = new EVector<T>[++no]; n=no; sz=0; };
    ~EMatrix() { delete[] ar; };
    bool isEmpty(){ return sz == 0; };
    void inverse(T val) {
        int res = 0;
        int inv = ar[1]->inverse(val, 1, res);
        while (inv != 0 && res > 0) {
            cout << ((res & 0x3) - 1);
            res >>=2;
        }
        cout << " " << inv << endl;
    };
    void create(T values[], int n) {
        EVector<T> *el;
        ar[0] = NULL;
        for (int i=0; i<n; i++) {
            el = new EVector<T>(8*n);
            el->insert(values[i]);
            write(el);
        }
    }
};
```

```

void write(EVector<T> *el) {
    int tmp = ++sz;
    while (tmp > 1 && (*ar[tmp/2] > *el)) {
        ar[tmp] = ar[tmp/2];
        tmp /= 2;
    }
    ar[tmp] = el;
}

EVector<T>* read() {
    EVector<T>* res = ar[1];
    EVector<T>* last = ar[sz];
    ar[sz--] = NULL;
    long tmp = 1;
    while (2 * tmp < sz + 1) {
        long next = 2 * tmp;
        if (next + 1 < sz + 1 && *ar[next+1] < *ar[next])
            next++;
        if (!(*last > *ar[next])) break;
        ar[tmp] = ar[next];
        tmp = next;
    }
    ar[tmp] = last;
    return res;
}

void process() {
    EVector<T> *tmp = new EVector<T>(8*n);
    while (sz > 1) {
        EVector<T> *p1 = read();
        EVector<T> *p2 = read();
        tmp->copy(p1, false);
        tmp->copy(p2, true);
        write(tmp);
        tmp = p1->clear();
        delete p2;
    }
    delete tmp;
}

};

class Point {
    int x, y;
public:
    Point() { x = -1; y = 0; };
    Point(int k, int v) { x = k; y = v; };
    bool operator < (const Point& ass) { return y < ass.y; };
    bool operator > (const Point& ass) { return y > ass.y; };
    bool operator == (const Point& ass)
        { return x == ass.x && (x != -1 || y == 0); };
    const Point operator + (const Point& ass)
        { Point res; res.x=-1; res.y=y+ass.y; return res; };
};

void main() {
    const int n = 8, m = 12;
    unsigned char e[] = { 0x16, 0x25, 0x4B, 0x72, 0x82, 0xAB, 0xD9, 0xE5 };
    unsigned char f[] = { 0xA2, 0x45, 0xCD, 0x38, 0x59, 0xFB, 0x84, 0x24,
                          0x7A, 0x0F, 0x15, 0xBD };

    Point a[8], b[12];
    for (int i=0; i<n; i++)
        a[i] = Point(e[i]&0xE0, e[i]&0x1F);
    for (int i=0; i<m; i++)
        b[i] = Point(f[i]&0xE0, f[i]&0x1F);
    EMatrix<Point> mat(n);
    mat.create(a, n);
    mat.process();
    for (int i=0; i<m; i++)
        mat.inverse(b[i]);
}

```

Rešenja zadataka:

Task1 (16 points):

3KopA
3KopB
JadaA
JadaB
3KopB
3KopB
JadaA
JadaB
3KopB
9000
9000KopA
9000KopB
9000KopB
81000000

Task2 (16 points):

kop
kop
ELE
kop
ELE
ELE
1814
2318

Task3 (16 points):

11.8
Kop2 4
0.6
Kop2 14
2.4
Kop4 22
Kop1 23
Kop4 22
Kop1 23
Kop3 24
Kop2 29
Kop2 62
127

Task4 (16 points):

0 conC&D
1 conC&D
2 conC&D
3 conC&D
3 conC&D
3 desC&D
1 conC&D
2 conC&D
3 conC&D
4 conC&D
4 desC&D
5 desC&D
5 desC&D
5 conC&D
5 desC&D
4 desC&D

6 conC&D
6 desC&D
3 desC&D
2 desC&D
3 desC&D
2 desC&D
1 desC&D
1 desC&D
0 desC&D

Task5 (16 points):

2 1
1 3
2 5
1 7
3 8
5 7
7 6
8 4
7 2
9 1
11 0
13 1
12 3
11 5
13 6
12 8

Task6 (18 points):

000
001
000
001
010
011
001
010

(10 10 20 20) (15 15 45 25)
(15 15 45 25)
(25 30 35 45)
(40 35 50 70)
(15 35 25 50) (20 40 30 55) (25 30 35 45)
(20 40 30 55) (25 30 35 45)
(0 60 25 80) (15 70 20 75)
(15 70 20 75)

Task7 (18 points):

010 4
011 4
10 3
11011 6
011 4
1100 5
11010 6
11011 6
111 4
00 3
00 3
010 4