

## Object oriented programming – Elektrijska 2013 (Balaton)

Task1 (14):					Task4 (14):		
Task2 (14):					Task5 (16):		
Task3 (14):					Task6 (14):		
Task1	Task2	Task3	Task4	Task5	Task6	Task7	Sum()

### **Task1. What is the output for the following program?**

```
#include <iostream>
#include <iomanip>
#include <math.h>

#define d -0.00126448926734961868021375957764

using namespace std;

double a = 0;
double r = 1;
double grrr = 22;

int n = 3, h0 = 256, equalCoo = 0;
static int mX,mY,mH;

class DrawA; class DrawB; class DrawC; class DrawD;
class Draw{
public:
    Draw(bool ord);
    ~Draw(){};
    void draw(int i, Draw& some);
    virtual void mainDraw(int level){};
protected:
    void equal(){if(mX == mY) {equalCoo++;cout << mX << endl;}}
    bool mOrd;
    int mLevel;
    double mSigned;
};
Draw::Draw(bool ord) : mOrd(ord) {};
void Draw::draw(int i, Draw &some){
    if (i > 0){
        cout<<setprecision(5)<<a<<" "<<r<<endl;
        if(mOrd){
            r = sqrt( (r *cos(a) + cos(mSigned))*(r *cos(a) + cos(mSigned))
                    + (r *sin(a)+sin(mSigned))*(r *sin(a)+sin(mSigned)) );
            a=(a+mSigned)/2;
            some.mainDraw(i-1);
        }else{
            r = sqrt( (r *cos(a) + cos(mSigned+grrr))*(r *cos(a) + cos(mSigned+grrr))
                    + (r *sin(a)+sin(mSigned+grrr))*(r *sin(a)+sin(mSigned+grrr)) );
            a=(a+mSigned+grrr)/2;
            some.mainDraw(i-1);
        }
    }
}
class DrawA : public Draw{
public:
    DrawA(bool ord) : Draw(ord){ mSigned = 0;}
    void mainDraw(int level);
};
class DrawB : public Draw{
public:
    DrawB(bool ord) : Draw(ord){mSigned = grrr/3;}
    void mainDraw(int level);
};
class DrawC : public Draw{
public:
    DrawC(bool ord) : Draw(ord){mSigned = 2 * grrr/3;}
    void mainDraw(int level);
};
```

```

void DrawA::mainDraw(int level){
    if (level&0x04){DrawC b(true);draw(level, b);}
    else{DrawC c(false);draw(level, c);}
}
void DrawB::mainDraw(int level){
    if (level&0x04){DrawA a(true);draw(level, a);}
    else{DrawA b(true);draw(level, b);}
}
void DrawC::mainDraw(int level){
    if (level&0x04){DrawB a(true);draw(level, a);}
    else{DrawB b(false);draw(level, b);}
}

int main(){
    int step = 0, hlen = h0, x0 = hlen / 2, y0 = x0, i = 0;
    i +=07;
    grrr = grrr/i+d;
    a = grrr-grrr/3+grrr;
    DrawA A(false);
    A.mainDraw(--i);
    cout << equalCoo << endl;
    return (int)0;
}

```

## **Task2. What is the output for the following program?**

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

int id = 0;
class A {
public:
    A() {
        id *= 2;
    }
    A(int objectNumber) {
        id += objectID = objectNumber;
        cout << objectID << " OOP" << endl;
    }
    ~A() {
        cout << id << " JADA " << id-- << endl;
    }
private:
    int objectID;
};

void create( void );
class B : public A {
public:
    B() : A(id) { static A b(id++); }
    B( int i): A(i) { A b(id); }
    ~B() { A b(id++); }
};

A a(id++);
class C : virtual public A {
    A a;
public:
    C(int i) : A(i) {}
    virtual ~C() { A b(); }
};

class D : public B, public C {
public:
    D(int i) : B(i), C(i) {
        static A a(i);
    }
};

int main( ) {
    C b(id);
    D c(id++);
    B d();
    create();
    return (int)0;
}

B b(1);
void create( void ) {
    static A b(id);
    C c(id++);
};
```

### **Task3. What is the output for the following program?**

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

int data = 0;
class A {
public:
    A() {data++;}
    A& operator++( ) {cout << "1 " << data-- << endl; A* b = new A; return *b;}
    A& operator++(int) {cout << "2 " << data++ << endl; A* b = new A; return *b;}
    A& operator--( ) {cout << "3 " << ++data << endl; A* b = new A; return *b;}
    A& operator--(int) {cout << "4 " << --data << endl; A* b = new A; return *b;}
    A operator+(A& b) { data++; return --b;}
    ~A() { }
};

class B {
public:
    virtual void f(A& c, int d = 2) = 0;
};

class C : public A, public B {
public:
    void f(A& c, int d = 5) { A a; A b=a+c; data+=d--;;}
};

void f(int i)
{
    C f;
    f.f(f,i%2);
    cout << data << endl;
    switch (i) {
        case 1: throw 1.2f; break;
        case 2: throw 3.2; break;
        case 3: throw A(); break;
        case 4: throw C(); break;
    }
}

int main()
{
    A i;
    B *j = new C();
    C l;
    int k = 1;
    f:
    try {
        for (; k<5; k++) f(k);
    }
    catch (A) {cout << "B";--i;}
    catch (C) {cout << "A";--i;}
    catch (double) {cout << "I";i--;}
    catch (float ) {cout << "L";++i;}
    if (k++<4) goto f;
    cout<<data;
    return (int)0;
}
```

#### **Task4. What is the output for the following program?**

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

enum Years { CCCXII, CCCXI, CCCX=310, CCCXIII=CCCX+CCCXII,
            MMXI, MMXII, MMXIII, MMX=2010 };

static int jubilee;
class Opa {
    int *pi;

public:
    int x;
    enum Years year;
    int jubilee() {cout << "Constantine the Great\n"; return x; };
    virtual int province();
    int jubilee(Opa& x);
    Opa(){jubile=0; x=3; year = MMXIII;};
    Opa(int i) : pi(new int(i)) {jubile++;}
    Opa(const Opa &x) : pi(new int(*x.pi)){this->x=313;}
    Opa& operator= (const Opa&);
};

Opa& Opa::operator= (const Opa& x) {
    if (this != &x) {
        delete pi;
        pi = new int(*x.pi);
    };
    return *this;
}

int Opa::jubilee(Opa& x){
    cout << "Edict of Milan " << year; this->x = x.jubilee() * x.province();
    cout << this->x <<endl; return this->x*2;
}

int Opa::province(){cout << "Mediana\n";return this->x*200+71;}

class Repa: public Opa{
public:
    virtual int jubilee();
    int jubilee(Opa& x);
    int province();
};

int Repa::jubilee(){ cout << "Year\n";return province()*10;}
int Repa::jubilee(Opa& x){ cout << year;return x.province()*10;}
int Repa::province(){ cout << "Naissus\n";return this->x *20;}

int main(){
    Years year = CCCXIII;
    Opa viminacium();
    Opa *roman = new Repa();
    Opa celebration;
    Repa emperor;

    cout << celebration.jubilee() + celebration.jubilee(emperor) << endl;
    cout << roman->jubilee() << celebration.province() << endl;

    cout << jubile*year << endl;
    return (int)0;
}
```

### **Task5. What is the output for the following program?**

```
#include <math.h>
#include <iostream>
using namespace std;

template <class Tip>
class Element {
public:
    Tip first, second;
    Element() { first = 0; second = -1; };
    Element(Tip f, Tip s) { first = f; second = s; };
    Element(const Element<Tip>& it) { first = it.first; second = it.second; };
    Element<Tip>& operator =(const Element<Tip>& it) {
        first = it.first; second = it.second;
        return *this;
    };
    bool operator !=(Element<Tip>& it) {
        return !(first == it.first && second == it.second);    };
    bool IsValid() { return second < 0; };
    Tip Square() { return (second - first+1)*(second + first)/2; };
    Tip Difference() { return second - first + 1; };
};

template <class Tip>
class Item {
public:
    int index;
    Tip value;
    Item<Tip>* pItem;
    Item<Tip>* pRel;
    bool status;
    Item(Item<Tip>* pI=NULL, Item<Tip>* pR=NULL, int ind=0, Tip val=Tip()) {
        pItem = pI; pRel = pR; index = ind; value = val; status = false;
    };
};

template <class Tip>
class Relationship {
    int size;
    Tip* arValues;
public:
    Item<Element<Tip> > *pItem;
    Relationship(Tip* arVal, int sz) {
        arValues = new Tip[sz];
        size = (int)sqrt((double)sz);
        for (int i=0; i<sz; )
            for (unsigned int j=0, val=0xF0000000; j<8; val>>=4, i++, j++)
                arValues[i] = (arVal[i/8] & val) >> (28-4*j);
        pItem = NULL;
    };
    void Accumulate() {
        Item<Element<Tip> >* pTemp = pItem;
        Tip sum = 0, count = 0, acc = 0, diff;
        for ( ; pTemp != NULL; pTemp=pTemp->pItem) {
            count += (diff = pTemp->value.Difference());
            sum += pTemp->value.Square();
            acc += diff * pTemp->index;
            cout<<pTemp->index<<" "<<pTemp->value.first<<" "<<pTemp->value.second<<endl;
        }
        cout<<count<<" "<<(1.0 * sum / count)<<" "<<(1.0 * acc / count)<<endl;
    };
    Element<Tip> Compute(Element<Tip>& pos, Tip limit) {
        Element<Tip> el(pos.first, pos.first);
        for ( ; el.first>=0 && arValues[el.first+pos.second*size]>=limit; el.first--);
        for ( ; el.second<size && arValues[el.second+pos.second*size]>=limit; el.second++);
        el.first++; el.second--;
        return el;
    };
};
```

```

void Create(Element<Tip> position, Tip limit) {
    Tip ind = position.second;
    Element<Tip> elPrev, el = Compute(position, limit);
    Item<Element<Tip> >* pItemTmp = new Item<Element<Tip> >(NULL, NULL, ind, el);
    Item<Element<Tip> >* pTemp1 = new Item<Element<Tip> >(NULL, pItemTmp),
        *pTemp2 = pTemp1;
    while (pTemp1 != NULL) {
        if (pItem == NULL)
            pItem = pItemTmp;
        else
            pItemTmp->pItem = pTemp1->pRel;
        pItemTmp = pTemp1->pRel;
        pTemp1 = pItemTmp->pItem;
        pTemp2 = pTemp1 ? pTemp2 : pTemp1;
        ind = pItemTmp->index;
        for (int i=0; i<2; i++) {
            Element<Tip> elPrev = el = pItemTmp->value;
            ind += i ? i+1 : i-1;
            while (ind >= 0 && ind < size && el.first <= elPrev.second) {
                el.second = ind;
                for(; arValues[el.first+ind*size]<limit && el.first<=elPrev.second;el.first++){
                    if (el.first <= elPrev.second) {
                        el = Compute(el, limit);
                        Item<Element<Tip> >* pR=pItemTmp->pRel;
                        for ( ; pR!=NULL && (pR->pItem->index != ind || pR->pItem->value != el); )
                            pR=pR->pRel;
                        if (pR == NULL) {
                            Item<Element<Tip> >* pItemNew=new Item<Element<Tip> >(NULL,NULL,ind,el);
                            pItemTmp->pRel = new Item<Element<Tip> >(pItemNew, pItemTmp->pRel);
                            pItemNew->pRel = new Item<Element<Tip> >(pItemTmp, pItemNew->pRel);
                            if (pTemp1 == NULL) {
                                pTemp1 = pTemp2 = new Item<Element<Tip> >(NULL, pItemNew);
                            } else {
                                pTemp2->pItem = new Item<Element<Tip> >(NULL, pItemNew);
                                pTemp2 = pTemp2->pItem;
                            }
                        }
                        el.first = ++el.second;
                    }
                }
            }
        }
    }
};

int main()
{
    int n = 64;
    unsigned int arValues[] = { 0x02321300, 0x13454340, 0x04537241, 0x02237541,
                                0x04357352, 0x02463241, 0x05254521, 0x00312310};
    Relationship<unsigned int> rel((unsigned int *)arValues, n);
    Element<unsigned int> el(5,3);
    rel.Create(el, 4);
    rel.Accumulate();
    return (int)0;
}

```



### **Task6. What is the output for the following program?**

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

template <class Tip>
class Node {
public:
    Tip value;
    Node* pLeft;
    Node* pRight;
    Node* pParent;
    int status;
    Node(Tip val=0, Node* pL=NULL, Node* pR=NULL) {
        value = val; status = 0;
        pLeft = pL; pRight = pR; pParent = NULL;
    };
    void Connect(Node<Tip>* pL) {
        pRight = new Node<Tip>(0, pL, pRight);
    };
    Node<Tip>* Find(Tip val) {
        Node<Tip>* pNode = this;
        for ( ; pNode!=NULL && val!=pNode->value; pNode=pNode->pLeft);
        return pNode;
    };
};

template <class Tip, int n>
class Forest {
public:
    int size;
    Node<Tip>* arTree[n];
    Forest(Tip* aVal, int nVal) {
        size = 0;
        for (int i=0; i<n; i++)
            arTree[i] = NULL;
        Node<Tip> *pTree=NULL, *pI, *pII;
        for (int i=0; i<nVal; i+=2) {
            if ((pI = pTree->Find(aVal[i])) == NULL) {
                pI = pTree = new Node<Tip>(aVal[i], pTree);
                size++;
            }
            if ((pII = pTree->Find(aVal[i+1])) == NULL) {
                pII = pTree = new Node<Tip>(aVal[i+1], pTree);
                size++;
            }
            pI->Connect(pII);
        }
        arTree[0] = pTree;
    };
    void Print(Node<Tip>* pNode) {
        for (cout << endl; pNode != NULL; cout << endl, pNode=pNode->pLeft) {
            cout << pNode->value;
            if (pNode->pParent == NULL)
                cout << " " << pNode->status;
            else
                cout << " " << pNode->pParent->value;
            for (Node<Tip>* pTmp=pNode->pRight; pTmp!=NULL; pTmp=pTmp->pRight) {
                cout << " " << pTmp->pLeft->value;
            }
        }
    };
};
```

```

Node<Tip>* Rotate(Node<Tip>* pTree) {
    Node<Tip>* pNew = new Node<Tip>(pTree->value, NULL), *pT = pNew;
    for (Node<Tip>* pNode=pTree->pLeft; pNode != NULL;
        pNode=pNode->pLeft, pT=pT->pLeft) {
        pT->pLeft = new Node<Tip>(pNode->value, NULL);
    }
    for (Node<Tip>* pNode=pTree; pNode != NULL; pNode=pNode->pLeft) {
        Node<Tip>* pI = pNew->Find(pNode->value);
        for (Node<Tip>* pTmp=pNode->pRight; pTmp!=NULL; pTmp=pTmp->pRight) {
            Node<Tip>* pII = pNew->Find(pTmp->pLeft->value);
            pII->Connect(pI);
        }
    }
    return pNew;
};

void Arrange(int ind) {
    arTree[ind+1] = Rotate(arTree[ind]);
    Process(arTree[ind+1]);
    Print(arTree[ind+1]);
    for (Node<Tip>* pI=arTree[ind], *pII=arTree[ind+1]; pI!= NULL;
        pI=pI->pLeft, pII=pII->pLeft) {
        pI->status = -pII->status;
    }
    Correct(arTree[ind]);
    Print(arTree[ind]);
};

void Process(Node<Tip>* pTree, bool bAssign=false) {
    int cnt = 0;
    for (Node<Tip>* pNode=pTree; pNode != NULL; pNode=pNode->pLeft)
        if (pNode->status <= 0)
            Assign(pNode, cnt, bAssign ? pNode : NULL);
};

void Correct(Node<Tip>* pTree) {
    int cnt = 0;
    int state = -size;
    for (Node<Tip>* pI=pTree, *pII; pI!=NULL; pI=pI->pLeft, state++) {
        for (pII=pTree; pII!=NULL && pII->status!=state; pII=pII->pLeft);
        if (pII != NULL)
            Assign(pII, cnt, pII);
    }
};

void Assign(Node<Tip>* pNode, int& cnt, Node<Tip>* pPar) {
    pNode->status = 1;
    if (pPar != NULL)
        pNode->pParent = pPar;
    for (Node<Tip>* pTmp=pNode->pRight; pTmp!=NULL; pTmp=pTmp->pRight)
        if (pTmp->pLeft->status <= 0)
            Assign(pTmp->pLeft, cnt, pPar);
    pNode->status = ++cnt;
};

int main()
{
    char aVal[22] = { 'P','D', 'A','M', 'H','A', 'M','H', 'K','E', 'K','P',
                     'D','I', 'I','P', 'M','V', 'V','K', 'E','V'};
    Forest<char, 3> forest((char*)aVal,22);
    forest.Arrange(0);
    return (int)0;
}

```

### **Task7. What is the output for the following program?**

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

#define ISTRIANGLE(a,b,c) (a - b)>c ? (a - c)>b ? (b - c)>a ? 1 : 2 : 2 : 2

int noTriangA = 0, noTriangB =0, noTriangC=0;

class A {
public:
    int i, j, k;
    A(int l) : k(l), j(k+2), i(j+4) {}
};

class B: public A {
public:
    B( ) : A(2) { }
};

class C: virtual public A {
public:
    C( ) : A(3) { }
};

int main() {
    C yy;
    B xx ;
    A zz = yy;
    A tt = xx;
    int s = 0;
    for(int i=0;i<10;i++)
    {
        if (xx.i == (xx.j + 1)){
            s+=3;
        }
        else {
            s+=3; }
        if (yy.i == (yy.j + 1)){
            s+=12;
        }
        else {
            s+=30; }
        if (zz.i == (zz.j + 1)){
            s+=114;
        }
        else {
            s+=300; }
        if (tt.i == (tt.j + 1)){
            s+=-5;
        }
        else {
            s+=-3; }
        noTriangA+= ISTRIANGLE(xx.i+yy.i,xx.j+yy.j,xx.k+yy.k);
        noTriangB+= ISTRIANGLE(xx.i+tt.i,xx.j+tt.j,xx.k+tt.k);
        noTriangC+= ISTRIANGLE(tt.i+yy.i,tt.j+yy.j,tt.k+yy.k);
        xx.i = yy.i;
        yy.i = zz.i;
        zz.i = tt.i;
        xx.j = yy.j;
        yy.j = zz.j;
        zz.j = tt.j;
        cout << " " << s << " tA=" << noTriangA<< " tB=" << noTriangB
            << " tC=" << noTriangC << endl;
        }
    }
    return (int)0;
}
```

## **Solutions: Object oriented programming – Balaton 2013**

### **Task1.**

5.236 1  
4.1888 1  
3.1416 1  
2.0944 1  
1.0472 1  
3.1416 1  
0

### **Task2.**

0 OOP  
1 OOP  
2 OOP  
3 JADA 4  
3 OOP  
12 OOP  
38 OOP  
75 JADA 76  
12 OOP  
162 OOP  
324 OOP  
1297 JADA 1298  
1296 JADA 1297  
1295 JADA 1296  
1295 OOP  
2590 JADA 2591  
2589 JADA 2590  
2588 JADA 2589  
2587 JADA 2588  
2586 JADA 2587  
2585 JADA 2586  
2584 JADA 2585  
2584 OOP  
5168 JADA 5169  
5167 JADA 5168  
5166 JADA 5167

### **Task3.**

3 7  
9  
L1 9  
3 13  
14  
I4 13  
3 18  
20  
B3 22  
3 27  
28  
B3 30  
31

#### **Task4.**

Constantine the Great  
Edict of Milan 313Constantine the Great  
Naissus  
180  
363  
Mediana  
Constantine the Great  
336071  
0

#### **Task5.**

3 4 6  
2 4 4  
2 6 6  
4 3 4  
4 6 6  
1 2 4  
1 6 6  
5 2 3  
5 6 6  
2 1 2  
6 3 5  
20 3.95 3.3

#### **Task6.**

V 6 M E  
I 9 D  
E 5 K  
K 4 V  
H 1 M  
M 3 A  
A 2 H  
D 8 P  
P 7 K I

V V K  
I I P  
E V V  
K V P E  
H M A  
M M V H  
A M M  
D I I  
P I D

#### **Task7.**

330 tA=2 tB=2 tC=2  
660 tA=4 tB=4 tC=4  
990 tA=6 tB=6 tC=6  
1320 tA=8 tB=8 tC=8  
1650 tA=10 tB=10 tC=10  
1980 tA=12 tB=12 tC=12  
2310 tA=14 tB=14 tC=14  
2640 tA=16 tB=16 tC=16  
2970 tA=18 tB=18 tC=18  
3300 tA=20 tB=20 tC=20