

Object-oriented programming

Task1 (15):	Task5 (15):
Task2 (15):	Task6 (15):
Task3 (10):	Task7 (15):
Task4 (15):	

Task1	Task2	Task3	Task4	Task5	Task6	Task7	Sum

Task1. What is the output for the following program?

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

class A
{
public:
    float x;

public:
    A(): x(1){}
    void fun(int i) { x+=i; cout << "avg";}
    virtual void fun(float i) { x*=i; cout << "bga"; }
    void fun1(int i) {x*=i+2;}
    void fun1(int i, float j) { x*=i*j; cout << "otr" << endl;}
    virtual void fun1(float i, float j) {x-=i*j; cout << "osm" << endl;}
    bool operator&&( A & rh) {
        return (x++ > (rh.x)--) || (++x > (rh.x)++);
    }
};

class B : public A
{
public:
    void fun(float) { fun1(2,3); x = x * 10; cout << "canj" ;}
    void fun1(float i, float j = 4) {x-=i*j; cout << "2010" ;}
};

void main()
{
    A a;
    B b;
    b.fun1(2,a.x);
    b.fun(1);
    a.fun(a.x);
    b.fun1(1);
    if(a && b)
        a.x +=7;
    cout << a.x << b.x << endl;
    exit(0);
}
```

Task 2. What is the output for the following program?

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

int id = 0;

class CandD {
public:
    CandD() { id = 1;}
    CandD(int objectNumber) {
        oID = objectNumber;
        cout << oID << " Canj" << endl;
    }
    ~CandD() { cout << id << " 2010" << endl;}
private:
    int oID;
};

void gen( void );

CandD a(id++);

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

class CD2 : virtual public CandD {
    CandD a;
public:
    CD2(int i) : CandD(i) {}
    virtual ~CD2() { CandD b(id++); }
};

class CD3 : virtual public CD1, public CD2 {
public:
    CD3(int i) : CD1(i), CD2(i) { static CandD a(i);}
};

void main( ) {
    CandD b(id++);
    static CandD c(id++);
    CD3 a(id);
    gen();
    CandD f(id++);
    exit(0);
}

void gen( void ) {
    CandD a(id++);
    static CandD b(id++);
    CandD c(id++);
}
```

Task 3. What is the output for the following program?

```
#include <iostream>

static int b;

class C {
    int *pi;
public:
    int a;
    int c;

    C(){b=0;};
    C(int i) : pi(new int(i)), c(b), a(b++) { b++; }
    C(const C &C) : pi(new int(*C.pi)) {b*=3;}
    C& operator= (const C&);
    ~C() {delete pi;}
};

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

C funF(C Ca)
{
    C Cnew(5);
    Ca = Cnew;
    std::cout << b << Cnew.a << Cnew.c ;
    return Ca;
}

void g() {

    C Ca=4, Cb=2;
    C Cc = Ca;
    Ca = funF(Cb);
    Cc = Ca;
}

int main(int argc, char* argv[])
{
    b=3;
    C d();
    C d1 = C(4);
    g();
    std::cout << b << d1.a << d1.c << std::endl;
    exit(0);
}
```

Task 4. What is the output for the following program?

```
#include <iostream>
#include <complex>

using namespace std;
#define MUL(a,b)  a*a

class Canj {
public:
    virtual void f( int );
    virtual void f( double );
    virtual void g( int i = 2010 );
};

void Canj::f( int i) { cout << "Canj" << i << endl; }

void Canj::f( double i) { cout << "CrnaGora" << i << endl; }

void Canj::g( int i ) {
    cout << i << endl;
    if (i==2010) { f(MUL(2+i,3-i));}
}

class CrnaGora: public Canj {
public:
    void f( complex<double> );
    void g( int i = 2011 );
};

void CrnaGora::f( complex<double> c) { cout << "Elektrijada" << c.real() << endl; }

void CrnaGora::g( int i ) {
    cout << "50" << i << endl;
    if (i==2010) { f(MUL(2+i,3-i));}
}

void f(Canj &a) { a.g(); }

void main() {
    Canj b;
    CrnaGora d;
    Canj* pb = new CrnaGora;
    b.f(3.0);
    d.f(4.0);
    pb->f(5.0);
    b.g();
    d.g();
    pb->g();
    f(d);
    delete pb;
    exit(0);
}
```

Task 5. What is the output for the following program?

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

template <class T>
class Array {
    T *ar;
    int n;
    void Swap(int a, int b) { T tmp=ar[a]; ar[a]=ar[b]; ar[b]=tmp; };
    int FindTarget(int a, int b);
    int Rewrite(int a, int b);
    T Write(int a, int b, int c);
public:
    void SetArray(T *arr, int nn);
    void Insert(T data);
    void Calc(int a);
    int Shift(int a);
    void Print(T val);
};

template <class T>
void Array<T>::SetArray(T *arr, int nn) {
    n = nn;    ar = new T[n];
    for (int i=0; i<n; i++) ar[i] = arr[i];
}

template <class T>
void Array<T>::Insert(T data) {
    T *arr = ar;
    ar = new T[++n];
    int i;
    for (i=n-1; i>0; i--)
        if (arr[i-1] > data) { ar[i] = arr[i-1]; break; }
    ar[--i] = data;
    for (--i; i>=0; i--)
        ar[i] = arr[i];
    delete[] arr;
}

template <class T>
T Array<T>::Write(int a, int b, int c) {
    if (b == c) return ar[b];
    int d = Rewrite(b, c);
    int e = d - b + 1;
    if (a == e) return ar[d];
    else if (a < e) return Write(a, b, d-1);
    else return Write(a-e, d+1, c);
}

template <class T>
int Array<T>::FindTarget(int a, int b) {
    int tar, min=100, sum;
    for (int i=a; i<b; i++) {
        T a = ar[i] | ar[i+1];
        for (sum=0; a; a>>=1)
            sum += a & 0x1;
        if (min > sum) { min = sum; tar = i; }
    }
    return tar;
}
```

```

template <class T>
int Array<T>::Rewrite(int a, int b) {
    int c = FindTarget(a, b);
    Swap(c, b);
    T x = ar[b];
    int d, e;
    for (d=a-1, e=a; e<b; e++)
        if (ar[e] <= x) Swap(++d, e);
    Swap(++d, b);
    return d;
}

template <class T>
int Array<T>::Shift(int a) {
    T str=0;
    for (int i=0; i<a; i++)
        str += ar[i];
    return str;
}

template <class T>
void Array<T>::Print(T val) {
    for (int i=0; i<8*sizeof(T); val>>=1, i++) {
        char b = (val & 0x1) + '0';
        cout << b;
    }
    cout << endl;
}

template <class T>
void Array<T>::Calc(int a) {
    T val;
    if (a <= 1)
        val = Write(a, 0, n-1);
    else {
        Calc(a / 2);
        val = Write(a, 0, n-1);
    }
    Print(val);
}

int main () {
    char n=7, ar[] = {'g', 'D', 'Z', 'a', 'i', '5', 'l'};
    Array<char> a;
    a.SetArray(ar, n);
    a.Calc(n/2);
    cout << a.Shift(n/2) << endl;
    a.Insert('P'); a.Insert('8');
    a.Calc(n/2+1);
    cout << a.Shift(n/2+1) << endl;
    return 0;
}

```

Task 6. What is the output for the following program?

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

class Basic {
public:
    int no;
public:
    Basic() {no = 0;};
    Basic(int num) {no = num;};
    Basic(Basic &b) { no=b.no; cout<<"Basic"<<endl; };
    Basic& operator=(Basic &b) { no=b.no; return *this; };
    virtual ~Basic() {};
    virtual bool IsGreater(Basic* pB) { return no > pB->no; };
    virtual void Write() { cout << no << " ";};
    virtual void Swap(Basic *b) { Basic tmp=*b; *b=*this; *this=tmp; }
};

class DerivedA : public Basic {
    int ind;
public:
    DerivedA() { ind=0; };
    DerivedA(int num) : Basic(num) { ind=0; };
    DerivedA(DerivedA &a) { no = a.no; cout<<"DerivedA"<<endl; };
    DerivedA& operator=(DerivedA &a) { no=a.no; ind++; return *this; };
    virtual ~DerivedA() {};
    virtual bool IsGreater(Basic* pB) {return no < pB->no; };
    virtual void Write() { cout << no << " " << ind << " ";};
    void Swap(Basic *b) {Swap((DerivedA *)b);};
    void Swap(DerivedA *a) { DerivedA tmp=*a; *a=*this; *this=tmp; }
};

class DerivedB : public Basic {
    int ind;
public:
    DerivedB() { ind=0; };
    DerivedB(int num) : Basic(num) { ind=0; };
    DerivedB(DerivedB &b) { no = b.no; cout<<"DerivedB"<<endl; };
    DerivedB& operator=(DerivedB &b) { no=b.no; ind++; return *this; };
    virtual ~DerivedB() {};
    bool IsGreater(Basic* pB) { return no < pB->no; };
    virtual void Write() { cout << no << " " << ind << " ";};
    void Swap(Basic *b) {Swap((DerivedB *)b);};
    virtual void Swap(DerivedB *b) { DerivedB tmp=*b; *b=*this; *this=tmp; }
};

class DerivedD : public DerivedB {
public:
    DerivedD() {};
    DerivedD(int num) : DerivedB(num) { };
    DerivedD(DerivedD &d) { no = d.no; cout<<"DerivedD"<<endl; };
    virtual ~DerivedD() {};
    bool IsGreater(Basic* pB) { return no > pB->no;};
    void Swap(Basic *b) {Swap((DerivedD *)b);};
    void Swap(DerivedD *d) { DerivedD tmp=*d; *d=*this; *this=tmp; }
};
```



```

int main()
{
    int i, n=5;
    Basic** aBasic = new Basic*[n];
    aBasic[0] = new DerivedA(4);
    aBasic[1] = new Basic(7);
    aBasic[2] = new Basic(3);
    aBasic[3] = new DerivedB(15);
    aBasic[4] = new DerivedD(11);
    for (i=0; i<n-1; i++) {
        int nMax = i;
        for (int j=i+1; j<n; j++)
            if (aBasic[j]->IsGreater(aBasic[nMax]))
                nMax = j;
        if (nMax != i)
            aBasic[nMax]->Swap(aBasic[i]);
    }
    for (i=0; i<n; i++)
        aBasic[i]->Write();
    cout << endl;
    for (i=0; i<n-1; i++) {
        int nMax = i;
        for (int j=i+1; j<n; j++)
            if (!aBasic[nMax]->IsGreater(aBasic[j]))
                nMax = j;
        if (nMax != i)
            aBasic[i]->Swap(aBasic[nMax]);
    }
    for (i=0; i<n; i++)
        aBasic[i]->Write();
    cout << endl;
    return 0;
}

```

Task 7. What is the output for the following program?

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

class Node {
public:
    int val;
    int code;
    Node *down, *right, *prev;
    Node() {down=right=prev=NULL; val=code=0;};
};

class Path {
    Node *start;
public:
    Path() { start = NULL; };
    void Traverse(char *arin, int nin);
    char* Process(char* ar);
    int Length(Node *pNode, char c);
};

class Net {
public:
    Node **ar;
    int ind, n;
    Net() { ar=NULL; ind=0; n=0;};
    void Init(Node *arin, int nin);
    void Connect();
    Node* GetLast() { return ar[n-1]; };
};

void Net::Init(Node *arin, int nin) {
    int i, j;
    ar = new Node*[n=nin];
    for (i=0; i<nin; i++) ar[i] = &arin[i];
    for (i=0; i<nin-1; i++)
        for (j=nin-1; j>i; j--)
            if (ar[j]->val<ar[j-1]->val) {
                Node *pNode=ar[j];
                ar[j]=ar[j-1];
                ar[j-1]=pNode;
            }
}

void Net::Connect() {
    Node *pNode = new Node();
    pNode->down = ar[ind];
    ar[ind]->prev = pNode;
    pNode->val = ar[ind]->val + ar[++ind]->val;
    pNode->right = ar[ind];
    ar[ind]->prev = pNode;
    ar[ind] = pNode;
    int i = ind-1;
    while (++i<n-1 && pNode->val>ar[i+1]->val)
        ar[i] = ar[i+1];
    ar[i] = pNode;
}
```

```

void Path::Traverse(char *arin, int nin) {
    int i, j;
    Node *artmp = new Node[2*nin];
    int ntmp = 0;
    for (i=0; i<nin; i++) {
        for (j=0; j<ntmp && arin[i]!=artmp[j].code; j++);
        if (j < ntmp) {
            artmp[j].val++;
        } else {
            artmp[ntmp].val=1;
            artmp[ntmp++].code = arin[i];
        }
    }
    Net frst;
    frst.Init(artmp, ntmp);
    for (i=0; i<ntmp-1; i++)
        frst.Connect();
    start=frst.GetLast();
}

```

```

char* Path::Process(char *ar) {
    int j=1, k=0, len=0;
    char c=0, *str = new char[strlen(ar)];
    for (int i=0; i<strlen(ar); i++) {
        int val = Length(start, ar[i]);
        for (k=0; val!=1; k++, j++, c<=&1, val/=2) {
            c |= val%2;
            if (j%8==0) { str[len++] = c; j=0; c=0; }
        }
        cout << k << " ";
    }
    cout << endl;
    if (j>1) str[len++] = c<<(9-j);
    str[len] = '\0';
    for (i=0; i<strlen(str); i++) {
        c = str[i];
        for (j=0; j<8; j++)
            cout << ((c & (0x1<<(7-j))) ? 1:0);
    }
    return str;
}

```

```

int Path::Length(Node *pNode, char c) {
    if (!pNode->down && !pNode->right)
        if (pNode->code == c) return 1;
        else return -1;
    int val;
    if ((val=Length(pNode->down, c))>=0)
        return val * 2 + 1;
    else
        if ((val=Length(pNode->right, c))>=0)
            return val * 2;
    return -1;
}

```

```

int main()
{
    char* niz="013554540320894670584241067130";
    Path stablo;
    stablo.Traverse(niz, strlen(niz));
    stablo.Process("0038163");
    return 0;
}

```

Elektrijada 2010 - Čani

Solutions:

Task1 (15 points):

20102010canjbga20109-75

Task2 (15 points):

0 Canj
1 Canj
2 Canj
3 Canj
1 2010
3 Canj
1 Canj
2 Canj
3 Canj
4 2010
4 2010
4 Canj
5 2010
5 2010
5 2010
5 2010

Task3 (10 points):

83818224934

Task4 (15 points):

CrnaGora3
Elektrijada4
CrnaGora5
2010
Canj6032
502011
502010
Elektrijada6032
502010
Elektrijada6032

Task5 (15 points):

10101100
01011010
-45
10101100
00011100
00001010
1

Task6 (15 points):

DerivedD
DerivedD
11 1 7 4 15 0 3 2
DerivedA
Basic
Basic
DerivedB
3 2 11 7 4 1 15 4

Task7 (15 points):

2 2 3 4 4 4 3
111110101010000000101000