

Программирование

Лекция 5

1. Конструктор копирования.
2. Динамическое выделение памяти.
3. Композиция.

Конструктор копирования

- Конструктор копирования позволяет решить проблемы, возникающие при передаче объектов в функции по значению и возвращении объектов функциями.
- **Суть проблемы:**
 - 1. Если объект передается в функцию по значению, то в функции создается поразрядная **копия объекта**. Если исходный объект содержит указатель на динамически выделенный блок памяти, то в копии этот указатель будет хранить адрес **того же** блока памяти. Если функция изменит содержимое динамически выделенной памяти, то он **изменится** и для исходного объекта.
После завершения работы функции деструктор может освободить общую для этих объектов область памяти.

Конструктор копирования

- Суть проблемы:
 - 2. Если функция возвращает объект, то для этого создается **временный** объект для хранения возвращаемого значения.
После завершения функции для временного объекта вызывается **деструктор**, а затем объект уничтожается.
Если деструктор удалит динамически выделенный блок памяти, то произойдет сбой.
 - Конструктор копии позволяет переопределить механизм копирования объекта так, чтобы избежать указанных проблем.

Конструктор копирования

- Конструктор копии вызывается, когда происходит инициализация другим объектом:
 - При создании объекта его инициализируют другим объектом.
 - Объект передается в функцию по значению.
 - Функция возвращает объект.
 - Конструктор копии записывается как конструктор с параметром – ссылкой на объект того же класса.

Конструктор копирования. Пример 1

```
class Test {  
    int x;  
  
public:  
    Test() {  
        x = 0;  
        cout << "Constructor, " << x << endl;  
    }  
  
    Test(int v) {  
        x = v;  
        cout << "Constructor param, " << x << endl;  
    }  
}
```

Конструктор копирования. Пример 1

```
Test(Test& t) {  
    x = t.x;  
    cout << "Copy constructor, " << x << endl;  
}
```

```
~Test() {  
    cout << "Destructor, " << x << endl;  
}
```

```
void print() {  
    cout << "Print, " << x << endl;  
}
```

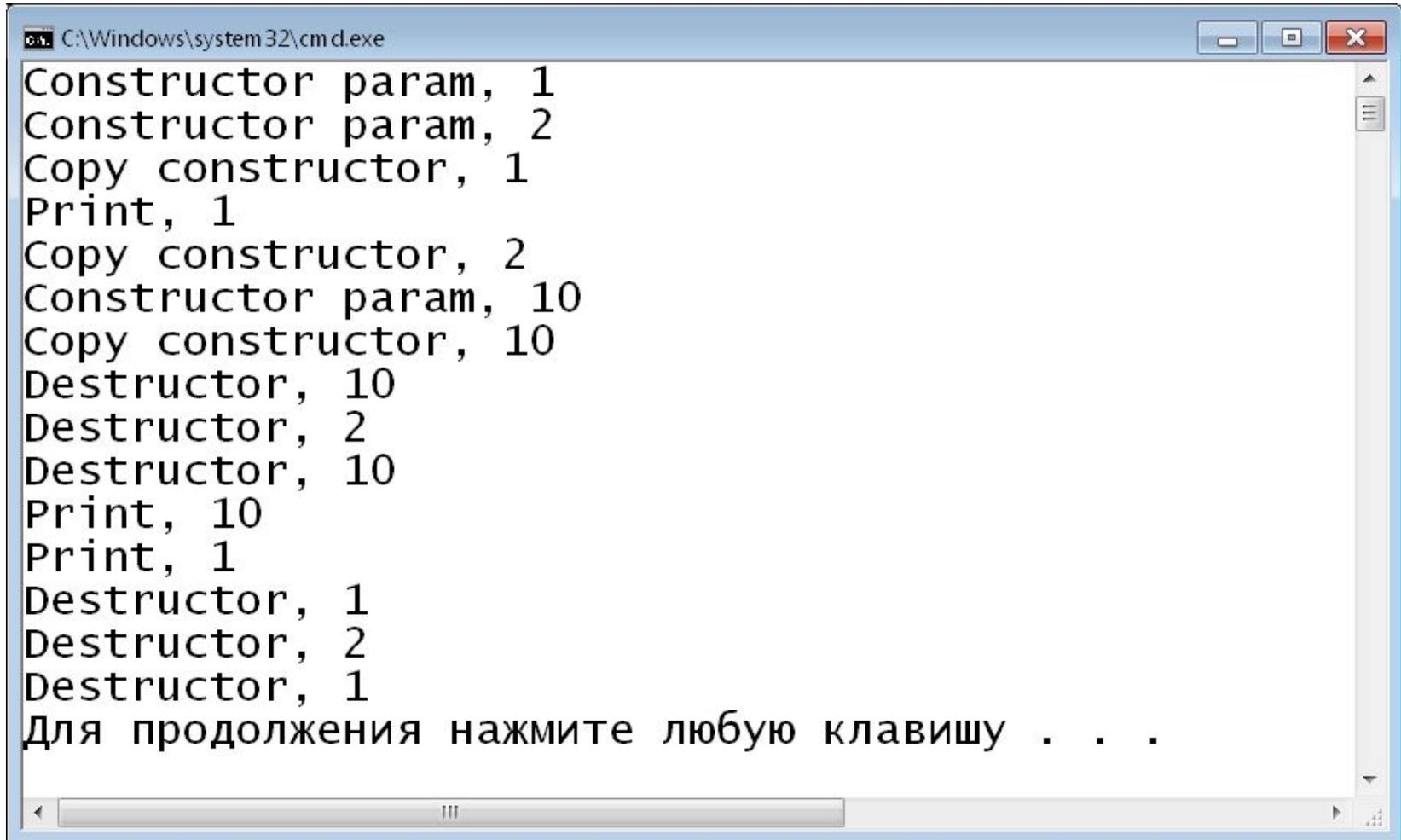
```
};
```

```
Test Func(Test t) {  
    Test t1(10);  
    return t1;  
}
```

Конструктор копирования. Пример 1

```
int main() {  
    Test a(1), b(2);  
  
    Test c = a;  
    c.print();  
  
    c = Func(b);  
    c.print();  
  
    c = a;  
    c.print();  
  
    return 0;  
}
```

Конструктор копирования. Пример 1



```
C:\Windows\system32\cmd.exe
Constructor param, 1
Constructor param, 2
Copy constructor, 1
Print, 1
Copy constructor, 2
Constructor param, 10
Copy constructor, 10
Destructor, 10
Destructor, 2
Destructor, 10
Print, 10
Print, 1
Destructor, 1
Destructor, 2
Destructor, 1
Для продолжения нажмите любую клавишу . . .
```

Правило трёх

Если в классе или структуре определен один из следующих методов, то необходимо явным образом определить все три метода:

- Деструктор
- Конструктор копирования
- Оператор присваивания копированием

Конструктор копирования. Пример 2

```
class Array {
    int* p;
    int size;
    char* name;

public:
    Array(int sz, char* nm) {
        p = new int[sz];
        if(!p) return;
        size = sz;
        name = nm;
        cout << "Constructor, ";
        cout << name << endl;
    }

    ~Array() {
        delete []p;
        cout << "Destructor, ";
        cout << name << endl;
    }

    Array(Array& a);

    void put(int i, int j) {
        if(i >= 0 && i < size)
            p[i] = j;
    }

    int get(int i) {
        return p[i];
    }

    char* getName() {
        return name;
    }
};
```

Конструктор копирования. Пример 2

```
Array::Array(Array& a){
    p = new int[a.size];
    if(!p) return;
    for(int i = 0; i < a.size; i++)
        p[i] = a.p[i];
    size = a.size;
    name = "Copy";
    cout << "Copy constructor, ";
    cout << name << endl;
}
```

```
int main() {
    int i;
    Array num(10, "Num");

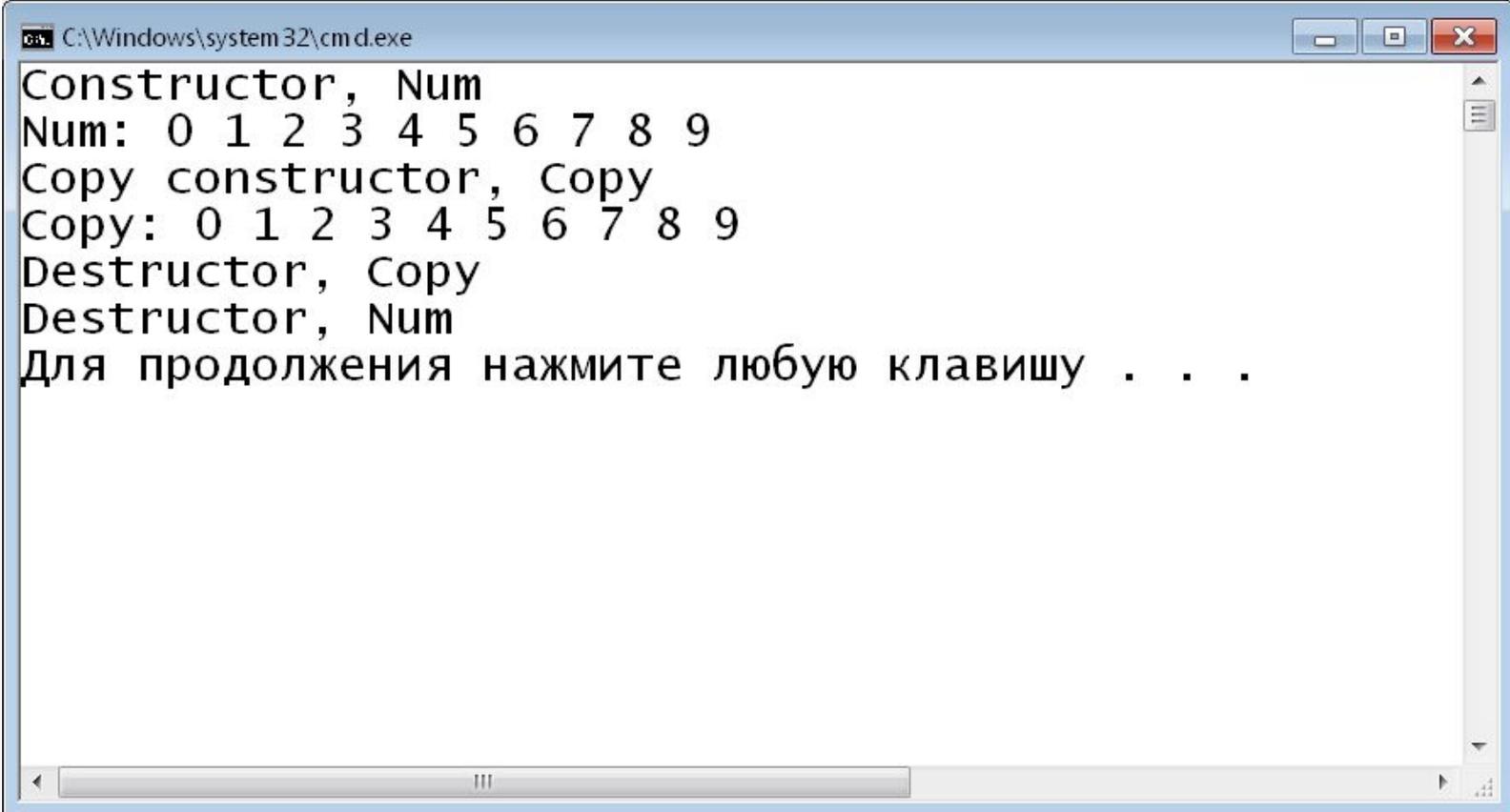
    for(int i = 0; i < 10; i++)
        num.put(i, i);

    cout << num.getName() << ": ";
    for(int i = 0; i < 10; i++)
        cout << num.get(i) << ' ';
    cout << endl;

    Array x = num;
    cout << x.getName() << ": ";
    for(int i = 0; i < 10; i++)
        cout << x.get(i) << ' ';
    cout << endl;

    return 0;
}
```

Конструктор копирования. Пример 2



```
C:\Windows\system32\cmd.exe
Constructor, Num
Num: 0 1 2 3 4 5 6 7 8 9
Copy constructor, Copy
Copy: 0 1 2 3 4 5 6 7 8 9
Destructor, Copy
Destructor, Num
Для продолжения нажмите любую клавишу . . .
```

Динамическое выделение памяти

```
class Samp {  
    int i, j;  
public:  
    Samp(int i = 0, int j = 0) {  
        Samp::i = i;  
        this -> j = j;  
    }  
  
    int product() {  
        return i*j;  
    }  
};
```

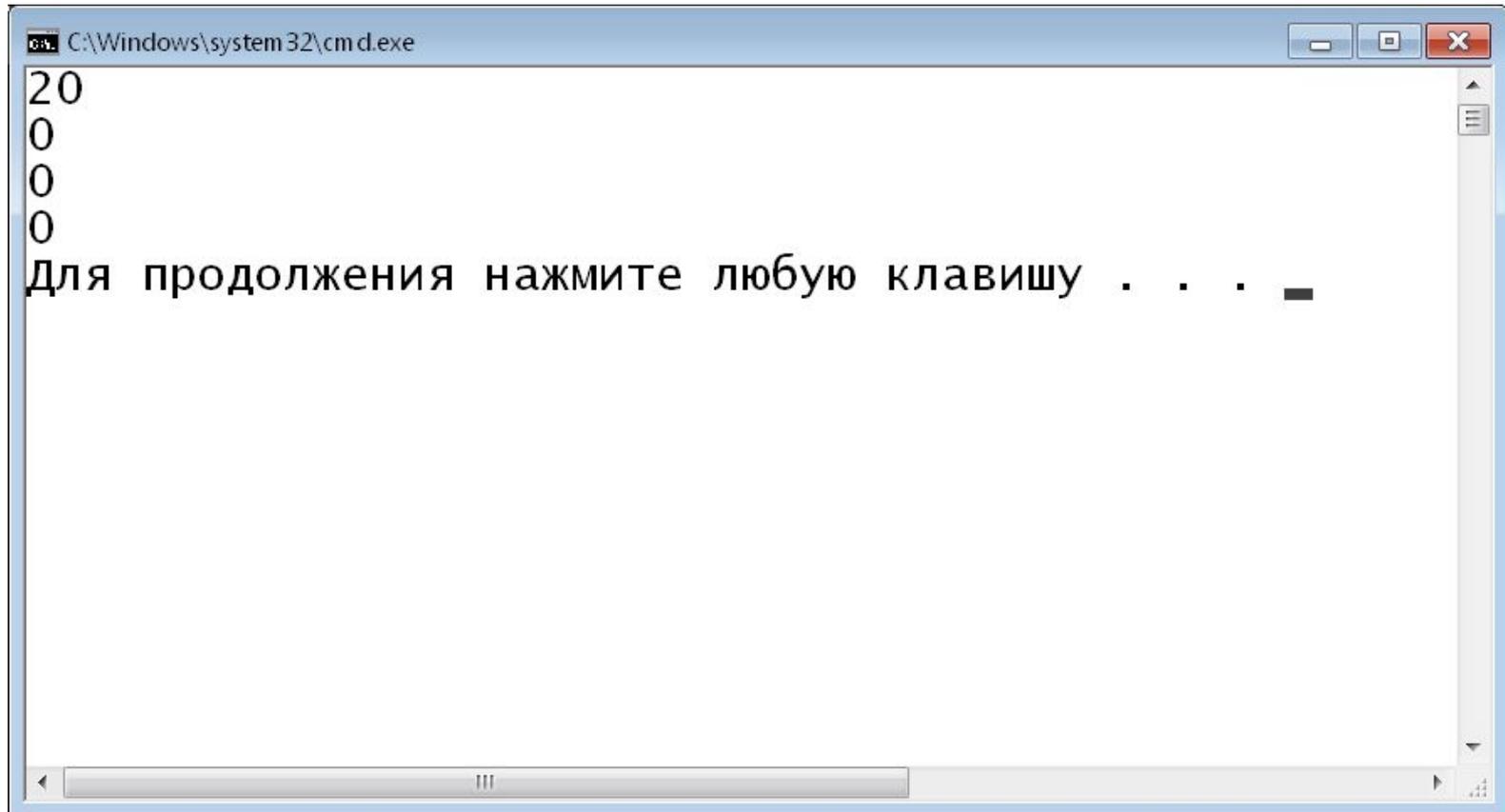
Динамическое выделение памяти

```
int main() {
    Samp* p;
    p = new Samp(4, 5);
    if(p == NULL) {
        cout << "Error!" << endl;
        return 1;
    }
    cout << p->product() << endl;

    Samp* m = new Samp[3];
    if(!m) {
        cout << "Error!" << endl;
        return 1;
    }
    for(int i = 0; i < 3; i++)
        cout << m[i].product() << endl;
    delete []m;

    return 0;
}
```

Динамическое выделение памяти



```
C:\Windows\system32\cmd.exe
20
0
0
0
Для продолжения нажмите любую клавишу . . .
```

The image shows a Windows command prompt window with a blue title bar. The title bar text is "C:\Windows\system32\cmd.exe". The window contains the following text: "20", "0", "0", "0", and "Для продолжения нажмите любую клавишу . . .". The text is displayed in a monospaced font. The window has standard Windows window controls (minimize, maximize, close) in the top right corner and a scrollbar on the right side.

Композиция

- Композиция – это метод связывания классов путем включения в класс объектов других классов в качестве данных-элементов.
- В этом случае классы связаны отношением «Часть – Целое».
- При создании объекта сложного класса сначала создаются объекты вложенных классов в порядке их объявления.

КОМПОЗИЦИЯ

```
class Date {
    int day, month, year;
    int check(int);
public:
    Date(int = 1, int = 1, int = 2015);
    void print();
};

Date::Date(int d, int m, int y) {
    year = y;
    if(m>0 && m<=12)
        month = m;
    else {
        cout << "Error in month: " << m << endl;
        month = 1;
    }
    day = check(d);
    cout << "Date: ";
    print();
}
```

КОМПОЗИЦИЯ

```
void Date::print() {
    cout << day << '-' << month << '-' << year << endl;
}

int Date::check(int x) {
    int days[13] = {0, 31, 28, 31, 30,
                   31, 30, 31, 31,
                   30, 31, 30, 31};
    if(x>0 && x<=days[month]) return x;
    if(x==29 && month == 2 &&
        (year%400 == 0 || (year%4 == 0 && year%100 != 0)))
        return x;
    cout << "Error in day: " << x << endl;
    return 1;
}
```

КОМПОЗИЦИЯ

```
class Worker {  
    char fam[25];  
    char name[25];  
    Date birth;  
    Date hire;  
public:  
    Worker(char*, char*,  
           int, int, int,  
           int, int, int);  
    void print();  
};
```

КОМПОЗИЦИЯ

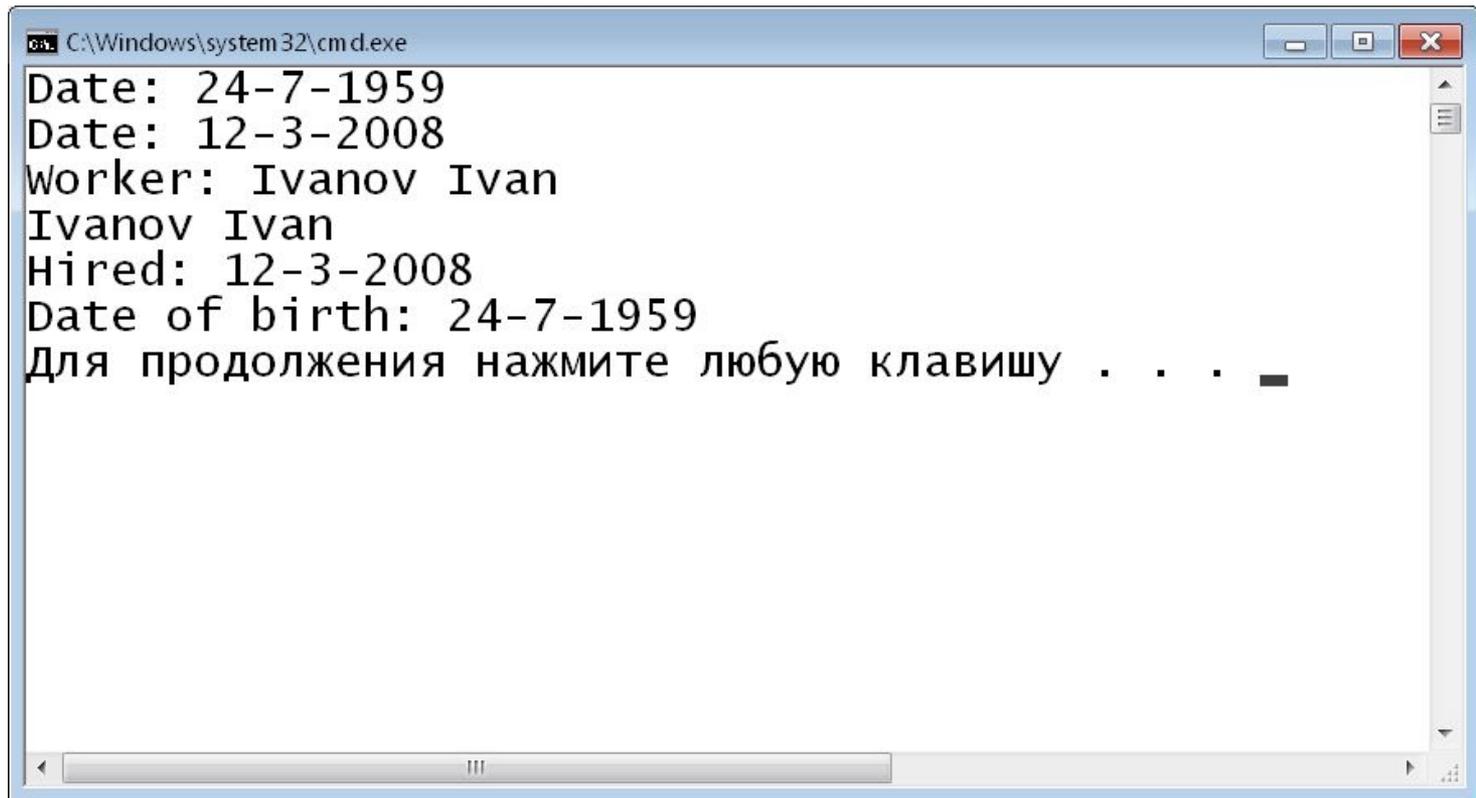
```
Worker::Worker(char* f, char* n,  
               int bd, int bm, int by,  
               int hd, int hm, int hy)  
    : birth(bd, bm, by),  
      hire(hd, hm, hy)  
{  
    int l;  
    l = strlen(f);  
    l=(l<25)?l:24;  
    strncpy(fam, f, l);  
    fam[l] = '\0';  
  
    l = strlen(n);  
    l=(l<25)?l:24;  
    strncpy(name, n, l);  
    name[l] = '\0';  
  
    cout << "Worker: " << fam << ' ' << name << endl;  
}
```

КОМПОЗИЦИЯ

```
void Worker::print() {  
    cout << fam << ' ' << name << endl;  
    cout << "Hired: ";  
    hire.print();  
    cout << "Date of birth: ";  
    birth.print();  
}
```

```
int main() {  
    Worker w("Ivanov", "Ivan",  
            24, 7, 1959,  
            12, 3, 2008);  
    w.print();  
    return 0;  
}
```

Композиция



A screenshot of a Windows command prompt window. The title bar shows the path `C:\Windows\system32\cmd.exe`. The window contains the following text output:

```
Date: 24-7-1959  
Date: 12-3-2008  
Worker: Ivanov Ivan  
Ivanov Ivan  
Hired: 12-3-2008  
Date of birth: 24-7-1959  
Для продолжения нажмите любую клавишу . . .
```

The text is displayed in a monospaced font. The window has a standard Windows interface with a title bar, minimize, maximize, and close buttons, and a scroll bar on the right side.