Repetitorium Test 3

Repetitorium Test 3

```
#include <iostream>
using namespace std;
class X {
  static int n;
  int no;
public:
  X(): no {n++} {cout << no;}
  X(const X& x): no {x.no*3} {}
  X& operator=(const X& x) {
    no = 2*x.no;
    return *this;
  }
  ~X() {cout << no;}
};
int X::n {5};

int main()
{
  X a, b, c{b = a};
  return 0;
}
```

```
#include <iostream>
using namespace std;
class X {
  static int n;
  int no;
public:
  X(): no {n++} {cout << no;}
  X(const X& x): no {x.no*5} {}
  X& operator=(const X& x) {
    no = 7*x.no;
    return *this;
  }
  ~X() {cout << no;}
};
int X::n {4};

int main()
{
  X a;
  {
    X b{a};
  }
  X d;
  return 0;
}
```

Repetitorium Test 3

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

class X {
  int a;
public:
  X(int n=9): a{n} {
    cout << a;
  }
  ~X() {
    cout << a;
  }
};

int main() {
  X y[3] {6, 2};
  {
    X *xp = new X[3] {4, 7};
  }
  return 0;
}
```

```
#include <iostream>
using namespace std;
class X {
  static int n;
  int no;
public:
  X(): no {n++} {cout << no;}
  X(const X& x): no {x.no*3} {}
  X& operator=(const X& x) {
    no = 5*x.no;
    return *this;
  }
  ~X() {cout << no;}
};
int X::n {2};

void f(const X& x1) {
  X *x2 {new X[2]};
}
int main()
{
  X a;
  f(a);
  return 0;
}
```

Repetitorium Test 3

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

class X {
protected:
  int x;
public:
  X(int i = 6): x {i} {}
  virtual void print() const { cout << x; }
  void operator-(const X rop) { this->print(); rop.print();}
};

class Y : public X {
  void print() const { cout << '=' << x; }
};

int main() {
  X x {9};
  Y y;
  x - y;
  y - x;
  return 0;
}
```

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

class X {
  int a;
public:
  X(int n = 7): a{n} {
    cout << a++;
  }
  X f() {
    return a+2;
  }
  ostream& print(ostream& o) { return o << a; }
};

int main() {
  X x, a {x.f()};
  a.print(cout);
  return 0;
}
```

Repetitorium Test 3

```
#include <iostream>
using namespace std;
class X {
  static int n;
  int no;
public:
  X(): no {n++} {cout << no;}
  X(const X& x): no {x.no*5} {}
  X& operator=(const X& x) {
    no = 8*x.no;
    return *this;
  }
  ~X() {cout << no;}
};
int X::n {4};

void f(const X& x1) {
  X x2 {x1};
}

int main()
{
  X a;
  f(a);
  return 0;
}
```

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

class X {
  int a;
public:
  X(int n = 3): a(n) { cout << a; }
  X(const X& x): a {x.a} {
    cout << 7*a;
  }
  ~X() {cout << a;}
  X& operator=(const X& x) {
    cout << a << x.a;
    return *this;
  }
};

void f(X a, const X& b) {
  a = b;
}

int main() {
  X x, y {5};
  f(x,y);
  return 0;
}
```

Repetitorium Test 3

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

class X {
public:
  int a;
  X(int n = 2): a {n} {
  }
  X(const X& x): a {x.a+6} {
    cout << a << x.a;
  }
};

int f(X x) {
  return x.a * 4;
}

int main() {
  X x;
  cout << f(x);
  return 0;
}
```

```
#include <iostream>
using namespace std;
class X {
  static int n;
  int no;
public:
  X(): no {n++} {cout << no;}
  X(const X& x): no {x.no*3} {}
  X& operator=(const X& x) {
    no = 7*x.no;
    return *this;
  }
  ~X() {cout << no;}
};
int X::n {2};

int main()
{
  X *ptr = nullptr;
  {
    X *a {new X};
    ptr = a;
  }
  X b, c;
  delete ptr;
  X d;
  return 0;
}
```

Repetitorium Test 3

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

class X {
  static int no;
public:
  X(int i = 8) {
    cout << ++no;
  }
  ~X() {
    cout << --no;
  }
};
int X::no {4};

int main() {
  X x, y {2};
  for (int i {0}; i<2; ++i) {
    X x {6};
  }
  X a;
  return 0;
}
```

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

class X {
  int a;
public:
  X(int n = 4): a{n} {
    cout << a;
  }
  X f() {
    X res {a+7};
    return res;
  }
  void print() {
    cout << "a:" << a;
  }
};

int main() {
  X x;
  x.f().print();
  return 0;
}
```

Repetitorium Test 3

```
#include <iostream>
using namespace std;
class X {
protected:
  static int n;
  int no;
public:
  X(): no {n++} {cout << no;}
  virtual void f(const X& x) {cout << "X" << no << x.no;}
};
int X::n {5};
class Y : public X {
public:
  void f(const X& x) {cout << "Y" << no;}
};

int main()
{
  X a, b;
  Y c;
  b = c;
  b.f(a);
  return 0;
}
```

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

class X {
  int a;
public:
  X(int n = 6): a {n} {}
  friend ostream& operator<<(ostream&, X);
};

ostream& operator<<(ostream& o, X x) {
  return  o << x.a+7;
}

int main() {
  X x {9};
  X y;
  y = x;
  cout << x << y;
  return 0;
}
```

Repetitorium Test 3

```
#include <iostream>
using namespace std;
class X {
protected:
  static int n;
  int no;
public:
  X(): no {n++} {cout << no;}
  virtual ostream& print (ostream& o) const  {return o << ':' << no;}
  void f(X x) const {
    this->print(cout);
    x.print(cout);
  }
};
int X::n {4};

class Y : public X {
public:
  ostream& print(ostream& o) const{X::print(o); return o << no+8;}
};

int main()
{
  Y a, b;
  a.f(b);
  return 0;
}
```