

0706 – merge STL

```
// merge STL
```

```
#include <iostream>
```

```
#include <algorithm>
```

```
#include <vector>
```

```
#include <iterator>
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int n, m;
```

```
    cin >> n >> m;
```

```
    vector<int> vA(n);
```

```
    vector<int>::iterator i;
```

```
    . i = vA.begin();
```

```
    . while (n > 0)
```

```
    . {
```

```
    .     cin >> *i;
```

```
    .     i++; n--;
```

```
    . }
```

```
    . merge(istream_iterator<int>(cin),
```

```
    . istream_iterator<int>(),
```

```
    . vA.begin(), vA.end(),
```

```
    . ostream_iterator<int>(cout, " "));
```

```
    .
```

```
    . return 0;
```

```
    .}
```

1303 Uppercase () ++

```
// 1303 Uppercase () ++
```

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

```
char l2up(char val1)
{ return
    val1 >= 'a' &&
    val1 <= 'z'
    ? val1 - 32
    : val1;
}
```

```
.int main()
.{
.
.   string vA;
.
.   std::getline (std::cin, vA);
.
.   ostream_iterator<char> out(cout);
.
.   transform(vA.begin(), vA.end(), out,
              l2up);
.
.   return 0;
.}
```

Перевод систем исчисления

- Дано число в системе исчисления по основанию A . Нужно перевести его в систему исчисления по основанию A^B .
-
- **Input**
- Со стандартного устройства ввода вводятся натуральные числа A и B и число L – длина записи числа в системе исчисления по основанию A . Гарантируется, что L кратно B и не превышает 10000. Во второй строке идет число N в системе исчисления A . N не превышает 10^9 . A^B не превышает 36 (то есть, всегда хватает цифр и латинских букв).
-
- **Output**
- Нужно выдать на стандартное устройство вывода число N в системе исчисления по основанию A^B .
-
- **Sample Input**
- 2 4 32
- 00111011100101110110011011001111
-
- **Sample Output**
- 3B9766CF

1304 – Перевод с/с

```
// 1304 – Перевод с/с
```

```
#include <iostream>
```

```
#include <algorithm>
```

```
#include <string>
```

```
#include <vector>
```

```
#include <iterator>
```

```
using namespace std;
```

```
char i2a_ (int val1)
```

```
{
```

```
    string s =
```

```
"0123456789ABCDEFGHIJKLMN  
OPQRSTUVWXYZ";
```

```
    return s[val1];
```

```
}
```

```
int main()
```

```
{
```

```
    string q, t;
```

```
    string::iterator ist;
```

```
    int a, b, n, s, k;
```

```
• cin >> a >> b >> n;
```

```
• vector<int> vA;
```

```
• vA.reserve(n / b);
```

```
•
```

```
• std::getline (std::cin, q);
```

```
• std::getline (std::cin, t);
```

```
•
```

```
• ist = t.begin();
```

```
• while ( ist < t.end() )
```

```
• {
```

```
•     s = *ist++ - 48;
```

```
•     for (k = 1; k < b; k++)
```

```
•         s = s * a + *ist++ - 48;
```

```
•     vA.push_back(s);
```

```
• }
```

```
• ostream_iterator<char> out(cout);
```

```
•
```

```
• transform( vA.begin(), vA.end(), out, i2a_ );
```

```
•
```

```
• return 0;
```

```
• }
```

Треугольник Паскаля

```
#include <iostream>
#include <algorithm>
#include <vector>
#include <iterator>
using namespace std;

int main()
{
    int n;
    cin >> n;
    vector<int> v(n);
    ostream_iterator<int> r(cout, " ");
    vector<int>::iterator end = v.begin();

    for (int i = 0; i < n; i++)
    {
        v[i] = 1;
        for (int j = i - 1; j > 0; j--)
            v[j] += v[j - 1];
        copy(v.begin(), end, r);
        cout << *end++ << endl;
    }

    return 0;
}
```

```
// transform () с обратным итератором !!!
#include <iostream>
#include <algorithm>
#include <vector>
#include <iterator>
using namespace std;

int sum ( int a, int b ) { return a + b; }

int main()
{
    int n, q;

    vector<int> vB;
    ostream_iterator<int> r(cout, " ");
    cin >> n;

    vB.push_back(1);
    cout << 1 << endl;
    for ( q=1; q < n; q++)
    {
        transform ( vB.rbegin(), vB.rend() - 1, vB.rbegin() + 1,
                    vB.rbegin(), sum );

        vB.push_back(1);
        copy(vB.begin(), vB.end()-1, r);
        cout << *(vB.end()-1) << endl;
    }
}
```