**EX1:**



#include<iostream>

#include<vector>

using namespace std;

template <class T>

bool palindrome(const vector<T>& target)

{

int left = 0, right = target.size() - 1;

while (left < right)

{

if (target[left] != target[right])

return false;

left++;

right--;

}

return true;

}

int main()

{

vector<int> target1;

vector<char> target2;

for (int i = 75; i >= 65; i--)

target1.push\_back(i);

for (vector<int>::iterator it = target1.begin(); it != target1.end(); it++)

cout << \*it << " ";

if (!palindrome(target1))

cout << " " << "is not a palindrome";

else cout << " " << "is a palindrome";

cout << endl;

for (char i = 'K'; i >= 'F'; i--)

target2.push\_back(i);

for (char i = 'G'; i <= 'K'; i++)

target2.push\_back(i);

for (vector<char>::iterator it = target2.begin(); it != target2.end(); it++)

cout << \*it << " ";

if (!palindrome(target2))

cout << " " << "is not a palindrome";

else cout << " " << "is a palindrome";

/\*vector<int> v1;

vector<int> v2;

for (int i = 0; i <= 100; i++)

v1.emplace\_back(i);

cout << v1.size() << " " << v1.capacity() << endl;

cout << v2.size() << " " << v2.capacity() << endl;

v1.swap(vector<int>());

cout << v1.size() << " " << v1.capacity() << endl;

cout << v2.size() << " " << v2.capacity() << endl;\*/

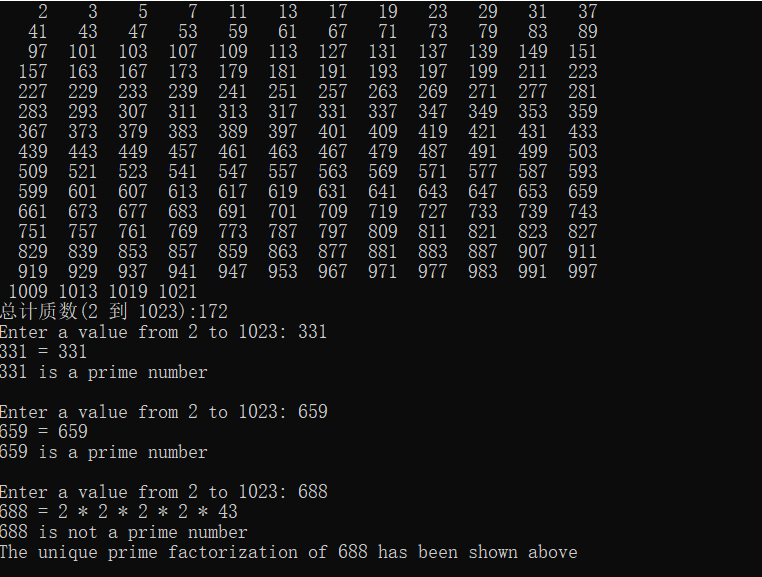
}

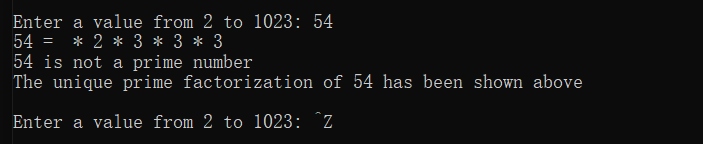
问题与思考：

Push\_back()也可以用Emplace\_back()来替换，更有效率。

定义迭代器的时候前面可以用c++11的auto关键字，虽然vs2010里也能成功，但它支持的11特性里似乎并没有auto这一项，挺奇特。

**EX2:**





#include<iostream>

#include<vector>

using namespace std;

bool PrimeFactors(const vector<bool>& target,int num)

{

static int cnt = 1;

bool isPrime = 0;

for (int i = 2; i <= 1023; i++)

{

if (target[i] == 0)

continue;

else

if (i == num)

{

if (cnt == 1) cout << i;

else cout << " \* " << i;

isPrime = 1;

return isPrime;

}

if (target[i] == 1 && num % i == 0)

{

if (cnt == 1) cout << i;

else cout << " \* " << i;

cnt++;

PrimeFactors(target, num / i);

return isPrime;

}

}

}

int main()

{

vector<bool> target(1024, 1);

for (int i = 2; i < 1023; i++) //1023本身不需要查找剩余

{

if(target[i] == 1)

for (int j = i + 1; j <= 1023; j++)

{

if (j % i == 0)

target[j] = 0;

}

}

int cnt = 0;

for (int i = 2; i <= 1023; i++)

if (target[i] == 1)

{

printf(" %4d",i), cnt++;

if (!(cnt % 12))

cout << endl;

}

cout << endl;

printf("总计质数(2 到 1023):%d\n", cnt);

int num = 0;

while (1)

{

printf("Enter a value from 2 to 1023: ");

if (cin >> num)

{

cout << num << " = ";

if (PrimeFactors(target, num))

printf("\n%d is a prime number\n\n", num);

else printf("\n%d is not a prime number\nThe unique prime factorization of %d has been shown above\n\n", num,num);

}

else break;

}

}

问题与思考：

这题跟STL好像一点关系都没有；

题目里面在递归中找到质因子我就当场输出了，没有选择装起来，使用了cin>>num里面cin状态的判断，如果输入CRTL+Z这样的cin会状态异常，然后退出循环。