Counting primes in a large range using block sieving technique

1. #include<bits/stdc++.h>
2. #define pp pop\_back
3. #define pb push\_back
4. #define int long long int
5. #define INF 1e18
6. #define vec vector<int>
7. #define pii pair<int,int>
8. #define REP(i,a,b) for(i=a;i<b;i++)
9. using namespace std;
10. vector<int> primes2;
11. void count\_primes(int n) {
12. const int S = 10000;
14. vector<int> primes;
15. int nsqrt = [sqrt](http://www.opengroup.org/onlinepubs/009695399/functions/sqrt.html)(n);
16. vector<char> is\_prime(nsqrt + 1, true);
17. for (int i = 2; i <= nsqrt; i++) {
18. if (is\_prime[i]) {
19. primes.push\_back(i);
20. for (int j = i \* i; j <= nsqrt; j += i)
21. is\_prime[j] = false;
22. }
23. }
25. vector<char> block(S);
26. for (int k = 0; k \* S <= n; k++) {
27. fill(block.begin(), block.end(), true);
28. int start = k \* S;
29. for (int p : primes) {
30. int start\_idx = (start + p - 1) / p;
31. int j = max(start\_idx, p) \* p - start;
32. for (; j < S; j += p)
33. block[j] = false;
34. }
35. if (k == 0)
36. block[0] = block[1] = false;
37. for (int i = 0; i < S && start + i <= n; i++) {
38. if (block[i])
39. primes2.pb(i+k\*S);
40. }
41. }
42. }
43. int32\_t main()
44. {
45. ios\_base::sync\_with\_stdio(false);
46. [cin](http://www.opengroup.org/onlinepubs/009695399/functions/cin.html).tie(NULL);
47. [cout](http://www.opengroup.org/onlinepubs/009695399/functions/cout.html).tie(NULL);
48. count\_primes(100000000);
49. int t=1;
50. cin>>t;
51. while(t--)
52. {
53. int n;
54. cin>>n;
55. auto it=lower\_bound(primes2.begin(),primes2.end(),n);
56. if((\*it)!=n)
57. cout<<"-1**\n**";
58. else
59. {
60. int q=it-primes2.begin()+1;
61. int p=([sqrt](http://www.opengroup.org/onlinepubs/009695399/functions/sqrt.html)(8\*q)-1)/2;
62. cout<<p+1<<" "<<q-(p\*(p+1))/2<<"**\n**";
63. }
64. }
65. }