1. *#include*<bits/stdc++.h>
3. typedef unsigned int uint;
4. typedef long long ll;
5. typedef unsigned long long ull;
6. typedef double lf;
7. typedef long double llf;
8. typedef std::pair<int,int> pii;
10. *#define* xx first
11. *#define* yy second
13. template<typename T> inline T max(T a,T b){return a>b?a:b;}
14. template<typename T> inline T min(T a,T b){return a<b?a:b;}
15. template<typename T> inline T abs(T a){return a>0?a:-a;}
16. template<typename T> inline bool repr(T &a,T b){return a<b?a=b,1:0;}
17. template<typename T> inline bool repl(T &a,T b){return a>b?a=b,1:0;}
18. template<typename T> inline T gcd(T a,T b){T t;if(a<b){while(a){t=a;a=b%a;b=t;}return b;}else{while(b){t=b;b=a%b;a=t;}return a;}}
19. template<typename T> inline T sqr(T x){return x\*x;}
20. *#define* mp(a,b) std::make\_pair(a,b)
21. *#define* pb push\_back
22. *#define* I inline
23. *#define* mset(a,b) memset(a,b,sizeof(a))
24. *#define* mcpy(a,b) memcpy(a,b,sizeof(a))
26. *#define* fo0(i,n) for(int i=0,i*##end=n;i<i##end;i++)*
27. *#define* fo1(i,n) for(int i=1,i*##end=n;i<=i##end;i++)*
28. *#define* fo(i,a,b) for(int i=a,i*##end=b;i<=i##end;i++)*
29. *#define* fd0(i,n) for(int i=(n)-1;~i;i--)
30. *#define* fd1(i,n) for(int i=n;i;i--)
31. *#define* fd(i,a,b) for(int i=a,i*##end=b;i>=i##end;i--)*
32. *#define* foe(i,x)for(\_\_typeof(x.end())i=x.begin();i!=x.end();++i)
34. struct Cg{I char operator()(){return getchar();}};
35. struct Cp{I void operator()(char x){putchar(x);}};
36. *#define* OP operator
37. *#define* RT return \*this;
38. *#define* RX x=0;char t=P();while((t<'0'||t>'9')&&t!='-')t=P();bool f=0;\
39. if(t=='-')t=P(),f=1;x=t-'0';for(t=P();t>='0'&&t<='9';t=P())x=x\*10+t-'0'
40. *#define* RL if(t=='.'){lf u=0.1;for(t=P();t>='0'&&t<='9';t=P(),u\*=0.1)x+=u\*(t-'0');}if(f)x=-x
41. *#define* RU x=0;char t=P();while(t<'0'||t>'9')t=P();x=t-'0';for(t=P();t>='0'&&t<='9';t=P())x=x\*10+t-'0'
42. *#define* TR \*this,x;return x;
43. I bool IS(char x){return x==10||x==13||x==' ';}template<typename T>struct Fr{T P;I Fr&OP,(int &x)
44. {RX;if(f)x=-x;RT}I OP int(){int x;TR}I Fr&OP,(ll &x){RX;if(f)x=-x;RT}I OP ll(){ll x;TR}I Fr&OP,(char &x)
45. {for(x=P();IS(x);x=P());RT}I OP char(){char x;TR}I Fr&OP,(char \*x){char t=P();for(;IS(t);t=P());if(~t){for(;!IS
46. (t)&&~t;t=P())\*x++=t;}\*x++=0;RT}I Fr&OP,(lf &x){RX;RL;RT}I OP lf(){lf x;TR}I Fr&OP,(llf &x){RX;RL;RT}I OP llf()
47. {llf x;TR}I Fr&OP,(uint &x){RU;RT}I OP uint(){uint x;TR}I Fr&OP,(ull &x){RU;RT}I OP ull(){ull x;TR}};Fr<Cg>in;
48. *#define* WI(S) if(x){if(x<0)P('-'),x=-x;char s[S],c=0;while(x)s[c++]=x%10+'0',x/=10;while(c--)P(s[c]);}else P('0')
49. *#define* WL if(y){lf t=0.5;for(int i=y;i--;)t\*=0.1;if(x>=0)x+=t;else x-=t,P('-');\*this,(ll)(abs(x));P('.');if(x<0)\
50. x=-x;while(y--){x\*=10;x-=floor(x\*0.1)\*10;P(((int)x)%10+'0');}}else if(x>=0)\*this,(ll)(x+0.5);else \*this,(ll)(x-0.5);
51. *#define* WU(S) if(x){char s[S],c=0;while(x)s[c++]=x%10+'0',x/=10;while(c--)P(s[c]);}else P('0')
52. template<typename T>struct Fw{T P;I Fw&OP,(int x){WI(10);RT}I Fw&OP()(int x){WI(10);RT}I Fw&OP,(uint x){WU(10);RT}
53. I Fw&OP()(uint x){WU(10);RT}I Fw&OP,(ll x){WI(19);RT}I Fw&OP()(ll x){WI(19);RT}I Fw&OP,(ull x){WU(20);RT}I Fw&OP()
54. (ull x){WU(20);RT}I Fw&OP,(char x){P(x);RT}I Fw&OP()(char x){P(x);RT}I Fw&OP,(const char \*x){while(\*x)P(\*x++);RT}
55. I Fw&OP()(const char \*x){while(\*x)P(\*x++);RT}I Fw&OP()(lf x,int y){WL;RT}I Fw&OP()(llf x,int y){WL;RT}};Fw<Cp>out;
57. struct \_ran
58. {
59. uint s;
61. \_ran(){s=19260817;}
63. inline uint operator()()
64. {
65. s^=s<<13;
66. s^=s>>17;
67. s^=s<<5;
68. return s;
69. }
70. }ran;
72. int startTime;
74. inline int realRunTime()
75. {
76. return clock()-startTime;
77. }
79. const int N=100007,M=1000007;
81. int n,m,K,type,s[N];
82. std::vector<int>res[M];
84. inline void gen(int t)
85. {
86. const int a[5]={10,100,990,9900,99000},b[5]={1,1,10,100,1000};
87. n=100000-ran()%1000;
88. fo0(i,n)s[i]=ran()%a[t]+b[t];
89. ll u=0;
90. fo0(i,n)u+=s[i];
91. *//m=min(u/2,1000000ll)-ran()%1000;*
92. m=1000000ll-ran()%1000;
93. fo0(i,m)res[i].clear();
94. type=t;
95. }
97. inline void input()
98. {
99. in,n,m;
100. fo0(i,n)in,s[i];
101. const int a[5]={10,100,1000,10000,100000};
102. type=0;
103. fo0(i,n)
104. {
105. int j=0;
106. for(;s[i]>a[j];j++);
107. repr(type,j);
108. }
109. }
111. inline void output()
112. {
113. out,K,'\n';
114. fo0(i,m/K)
115. {
116. out,(int)res[i].size();
117. foe(j,res[i])out,' ',\*j+1;
118. out,'\n';
119. }
120. }
122. inline ll score()
123. {
124. ll sum=0;
125. fo0(i,m/K)
126. {
127. ll su=0;
128. foe(j,res[i])su+=s[\*j];
129. if(su>K)return -1;
130. sum+=K-su;
131. }
132. *//out,K,' ',sum,'\n';*
133. return sum\*sum+(ll)K\*K;
134. }
136. namespace solve0
137. {
138. pii t[N];
140. inline ll chk(int l)
141. {
142. int v=l/2,ma=0;
143. fo0(i,v)repr(ma,t[i].xx+t[l-i-1].xx);
144. return(ll)ma\*v;
145. }
147. inline void solve()
148. {
149. fo0(i,n)t[i]=mp(s[i],i);
150. std::sort(t,t+n);
151. int l=0,r=n+1;
152. while(l+1<r)
153. {
154. int mid=(l+r)>>1;
155. if(chk(mid)<=m)l=mid;
156. else r=mid;
157. }
158. int v=l/2,ma=0;
159. *//out,n,' ',l,' ',r,' ',v,'\n';*
160. fo0(i,v)repr(ma,t[i].xx+t[l-i-1].xx);
161. K=ma;
162. fo0(i,v)res[i].pb(t[i].yy),res[i].pb(t[l-i-1].yy);
163. }
164. }
166. namespace solveType0
167. {
168. std::priority\_queue<pii>q;
169. pii t[N];
171. inline void solve()
172. {
173. fo0(i,n)t[i]=mp(s[i],i);
174. std::sort(t,t+n);
175. ll sum=0;
176. fo0(i,n)sum+=s[i];
177. if(sum>m)K=10;
178. else
179. {
180. ll best=sqr(m/10\*10-sum);
181. fo(i,2,10)
182. {
183. int v=m/i+1;
184. ll tmp=sqr((ll)v)+sqr(max(0ll,m/v\*v-sum));
185. if(tmp<best)K=v,best=tmp;
186. }
187. *//printf("%d %.2lf\n",K,sqrt(best));*
188. }
189. int cnt=m/K;
190. fo0(i,cnt)q.push(mp(K,i));
191. fd0(i,n)
192. {
193. if(!q.empty()&&q.top().xx>=t[i].xx)
194. {
195. pii v=q.top();q.pop();
196. res[v.yy].pb(t[i].yy);
197. if(v.xx-=t[i].xx)q.push(v);
198. }
199. }
200. }
201. }
203. namespace solveType1
204. {
205. std::priority\_queue<pii>q;
206. std::vector<int>g[N];
207. int gc[N];
208. pii t[N];
210. inline ll solveKo(int K)
211. {
212. int cnt=m/K,rem=K\*cnt;
213. fo0(i,cnt)q.push(mp(K,i));
214. fd0(i,n)
215. {
216. if(!q.empty()&&q.top().xx>=t[i].xx)
217. {
218. pii v=q.top();q.pop();
219. rem-=t[i].xx;
220. if(v.xx-=t[i].xx)q.push(v);
221. }
222. }
223. while(!q.empty())q.pop();
224. return(ll)K\*K+(ll)rem\*rem;
225. }
227. inline ll solveK(int K)
228. {
229. *//out,K,'\n';*
230. int now=0,u=0,cnt=m/K;
231. fo0(i,n)gc[s[i]]++;
232. for(int c=0;c<=K;c++)
233. {
234. for(int b=K-c;b>=0;b--)
235. {
236. while(b!=K-b-c?gc[b]&&gc[K-b-c]:gc[b]>1)
237. {
238. gc[b]--;
239. gc[K-b-c]--;
240. u+=c;
241. now++;
242. if(now==cnt)break;
243. }
244. if(now==cnt)break;
245. }
246. if(now==cnt)break;
247. }
248. u+=(cnt-now)\*K;
249. fo0(i,n)gc[s[i]]=0;
250. return(ll)u\*u+(ll)K\*K;
251. }
253. inline void solve()
254. {
255. fo0(i,n)t[i]=mp(s[i],i);
256. std::sort(t,t+n);
257. fo0(i,n)g[s[i]].pb(i);
258. ll best=1e18;
259. for(int i=1;i<N&&realRunTime()<CLOCKS\_PER\_SEC\*3.0;i++)
260. {
261. ll t=solveK(i);
262. if(t<best)best=t,K=i;
263. }
264. *//fo1(i,100)out,g[i].size(),'\n';*
265. *//exit(0);*
266. *//out,best,' ',K,'\n';*
267. */\*int cnt=m/K;*
268. *fo0(i,cnt)q.push(mp(K,i));*
269. *fd0(i,n)*
270. *{*
271. *if(!q.empty()&&q.top().xx>=t[i].xx)*
272. *{*
273. *pii v=q.top();q.pop();*
274. *res[v.yy].pb(t[i].yy);*
275. *if(v.xx-=t[i].xx)q.push(v);*
276. *}*
277. *}*
278. *while(!q.empty())q.pop();\*/*
279. int now=0,cnt=m/K;
280. for(int c=0;c<=K;c++)
281. {
282. for(int b=K-c;b>=0;b--)
283. {
284. while(b!=K-b-c?g[b].size()>0&&g[K-b-c].size()>0:g[b].size()>1)
285. {
286. *//if(now%1000==0)out,c,' ',b,' ',g[b].size(),' ',g[K-b-c].size(),' ',now,' ',cnt,'\n';*
287. res[now].pb(g[b].back());
288. g[b].pop\_back();
289. res[now].pb(g[K-b-c].back());
290. g[K-b-c].pop\_back();
291. now++;
292. if(now==cnt)break;
293. }
294. if(now==cnt)break;
295. }
296. if(now==cnt)break;
297. }
298. fo0(i,n)g[s[i]].clear();
299. }
300. }
302. namespace solveType2
303. {
304. std::priority\_queue<pii>q;
305. std::vector<int>g[N];
306. int gc[N];
307. pii t[N];
309. inline ll solveKo(int K)
310. {
311. int cnt=m/K,rem=K\*cnt;
312. fo0(i,cnt)q.push(mp(K,i));
313. fd0(i,n)
314. {
315. if(!q.empty()&&q.top().xx>=t[i].xx)
316. {
317. pii v=q.top();q.pop();
318. rem-=t[i].xx;
319. if(v.xx-=t[i].xx)q.push(v);
320. }
321. }
322. while(!q.empty())q.pop();
323. return(ll)K\*K+(ll)rem\*rem;
324. }
326. inline ll solveK(int K)
327. {
328. int now=0,u=0,cnt=m/K;
329. fo0(i,n)gc[s[i]]++;
330. for(int c=0;c<=K;c++)
331. {
332. for(int b=K-c;b>=0;b--)
333. {
334. while(b!=K-b-c?gc[b]&&gc[K-b-c]:gc[b]>1)
335. {
336. gc[b]--;
337. gc[K-b-c]--;
338. u+=c;
339. now++;
340. if(now==cnt)break;
341. }
342. if(now==cnt)break;
343. }
344. if(now==cnt)break;
345. }
346. u+=(cnt-now)\*K;
347. fo0(i,n)gc[s[i]]=0;
348. return(ll)u\*u+(ll)K\*K;
349. }
351. inline void solve()
352. {
353. fo0(i,n)t[i]=mp(s[i],i);
354. std::sort(t,t+n);
355. fo0(i,n)g[s[i]].pb(i);
356. ll best=1e18;
357. for(int i=1000,j=1001;realRunTime()<CLOCKS\_PER\_SEC\*3.0;i--,j++)
358. {
359. *//out,i,' ',j,'\n';*
360. ll t;
361. if(i>0)
362. {
363. t=solveK(i);
364. if(t<best)best=t,K=i;
365. }
366. t=solveK(j);
367. if(t<best)best=t,K=j;
368. }
369. *//out,best,' ',K,'\n';*
370. */\*int cnt=m/K;*
371. *fo0(i,cnt)q.push(mp(K,i));*
372. *fd0(i,n)*
373. *{*
374. *if(!q.empty()&&q.top().xx>=t[i].xx)*
375. *{*
376. *pii v=q.top();q.pop();*
377. *res[v.yy].pb(t[i].yy);*
378. *if(v.xx-=t[i].xx)q.push(v);*
379. *}*
380. *}*
381. *while(!q.empty())q.pop();\*/*
382. int now=0,cnt=m/K;
383. for(int c=0;c<=K;c++)
384. {
385. for(int b=K-c;b>=0;b--)
386. {
387. while(b!=K-b-c?g[b].size()>0&&g[K-b-c].size()>0:g[b].size()>1)
388. {
389. res[now].pb(g[b].back());
390. g[b].pop\_back();
391. res[now].pb(g[K-b-c].back());
392. g[K-b-c].pop\_back();
393. now++;
394. if(now==cnt)break;
395. }
396. if(now==cnt)break;
397. }
398. if(now==cnt)break;
399. }
400. fo0(i,n)g[s[i]].clear();
401. }
402. }
404. namespace solveType3
405. {
406. std::priority\_queue<pii>q;
407. std::set<pii>p;
408. std::vector<int>g[N];
409. int gc[N];
410. pii t[N];
412. inline ll solveK(int K)
413. {
414. int now=0,u=0,cnt=m/K;
415. fo0(i,n)gc[s[i]]++;
416. for(int c=0;c<=K;c++)
417. {
418. for(int b=K-c;b>=0;b--)
419. {
420. while(b!=K-b-c?gc[b]&&gc[K-b-c]:gc[b]>1)
421. {
422. gc[b]--;
423. gc[K-b-c]--;
424. u+=c;
425. now++;
426. if(now==cnt)break;
427. }
428. if(now==cnt)break;
429. }
430. if(now==cnt)break;
431. }
432. u+=(cnt-now)\*K;
433. fo0(i,n)gc[s[i]]=0;
434. return(ll)u\*u+(ll)K\*K;
435. }
437. inline ll solveK\_old(int K)
438. {
439. int cnt=m/K,rem=K\*cnt;
440. fo0(i,cnt)q.push(mp(K,i));
441. fd0(i,n)
442. {
443. if(!q.empty()&&q.top().xx>=t[i].xx)
444. {
445. pii v=q.top();q.pop();
446. rem-=t[i].xx;
447. if(v.xx-=t[i].xx)q.push(v);
448. }
449. }
450. while(!q.empty())q.pop();
451. return(ll)K\*K+(ll)rem\*rem;
452. }
454. inline ll solveK2(int K)
455. {
456. int cnt=m/K,rem=K\*cnt;
457. fo0(i,cnt)p.insert(mp(K,i));
458. fd0(i,n)
459. {
460. if(p.size()&&(--p.end())->xx>=t[i].xx)
461. {
462. if((--p.end())->xx>=t[i].xx+1)
463. {
464. *//out,"naive\n";*
465. pii v=\*--p.end();p.erase(v);
466. rem-=t[i].xx;
467. if(v.xx-=t[i].xx)p.insert(v);
468. }
469. else
470. {
471. pii v=\*p.lower\_bound(mp(t[i].xx,0));p.erase(v);
472. rem-=t[i].xx;
473. if(v.xx-=t[i].xx)p.insert(v);
474. }
475. }
476. }
477. p.clear();
478. return(ll)K\*K+(ll)rem\*rem;
479. }
481. inline void solve()
482. {
483. fo0(i,n)t[i]=mp(s[i],i);
484. std::sort(t,t+n);
485. fo0(i,n)g[s[i]].pb(i);
486. ll best=1e18;
487. for(int i=2500;i<=6500;i+=100)
488. {
489. ll t=solveK(i);
490. if(t<best)best=t,K=i;
491. }
492. for(int i=K,j=K+1;realRunTime()<CLOCKS\_PER\_SEC\*4.0;i--,j++)
493. {
494. ll t;
495. if(i>0)
496. {
497. t=solveK(i);
498. if(t<best)best=t,K=i;
499. }
500. if(j<9000)
501. {
502. t=solveK(j);
503. if(t<best)best=t,K=j;
504. }
505. }
506. *//out,K,' ',int(sqrt(best-K\*K)),' ',best,'\n';*
507. *//exit(0);*
508. */\*int cnt=m/K;*
509. *fo0(i,cnt)q.push(mp(K,i));*
510. *fd0(i,n)*
511. *{*
512. *if(!q.empty()&&q.top().xx>=t[i].xx)*
513. *{*
514. *pii v=q.top();q.pop();*
515. *res[v.yy].pb(t[i].yy);*
516. *if(v.xx-=t[i].xx)q.push(v);*
517. *}*
518. *}*
519. *while(!q.empty())*
520. *{*
521. *//out,q.top().xx,' ',q.top().yy,'\n';*
522. *q.pop();*
523. *}*
524. *fo0(i,cnt)*
525. *{*
526. *foe(j,res[i])out,s[\*j],' ';out,'\n';*
527. *}*
528. *while(!q.empty())q.pop();*
529. *fo0(i,n)g[s[i]].pb(i);\*/*
530. int now=0,cnt=m/K;
531. for(int c=0;c<=K;c++)
532. {
533. for(int b=K-c;b>=0;b--)
534. {
535. while(b!=K-b-c?g[b].size()>0&&g[K-b-c].size()>0:g[b].size()>1)
536. {
537. res[now].pb(g[b].back());
538. g[b].pop\_back();
539. res[now].pb(g[K-b-c].back());
540. g[K-b-c].pop\_back();
541. now++;
542. if(now==cnt)break;
543. }
544. if(now==cnt)break;
545. }
546. if(now==cnt)break;
547. }
548. fo0(i,n)g[s[i]].clear();
549. }
550. }
552. namespace solveType4
553. {
554. std::priority\_queue<pii>q;
555. std::set<pii>p;
556. std::vector<int>g[N];
557. int gc[N];
558. pii t[N];
560. inline ll solveK(int K)
561. {
562. int now=0,u=0,cnt=m/K;
563. fo0(i,n)gc[s[i]]++;
564. for(int c=0;c<=K;c++)
565. {
566. for(int b=K-c;b>=0;b--)
567. {
568. while(b!=K-b-c?gc[b]&&gc[K-b-c]:gc[b]>1)
569. {
570. gc[b]--;
571. gc[K-b-c]--;
572. u+=c;
573. now++;
574. if(now==cnt)break;
575. }
576. if(now==cnt)break;
577. }
578. if(now==cnt)break;
579. }
580. u+=(cnt-now)\*K;
581. fo0(i,n)gc[s[i]]=0;
582. return(ll)u\*u+(ll)K\*K;
583. }
585. inline void solve()
586. {
587. fo0(i,n)t[i]=mp(s[i],i);
588. std::sort(t,t+n);
589. fo0(i,n)g[s[i]].pb(i);
590. ll best=1e18;
591. for(int i=3000,j=3001;realRunTime()<CLOCKS\_PER\_SEC\*4.0;i--,j++)
592. {
593. *//out,i,' ',j,'\n';*
594. ll t;
595. if(i>0)
596. {
597. t=solveK(i);
598. if(t<best)best=t,K=i;
599. }
600. t=solveK(j);
601. if(t<best)best=t,K=j;
602. }
603. *//out,best,' ',K,'\n';*
604. */\*int cnt=m/K;*
605. *fo0(i,cnt)q.push(mp(K,i));*
606. *fd0(i,n)*
607. *{*
608. *if(!q.empty()&&q.top().xx>=t[i].xx)*
609. *{*
610. *pii v=q.top();q.pop();*
611. *res[v.yy].pb(t[i].yy);*
612. *if(v.xx-=t[i].xx)q.push(v);*
613. *}*
614. *}*
615. *while(!q.empty())q.pop();\*/*
616. int now=0,cnt=m/K;
617. for(int c=0;c<=K;c++)
618. {
619. for(int b=K-c;b>=0;b--)
620. {
621. while(b!=K-b-c?g[b].size()>0&&g[K-b-c].size()>0:g[b].size()>1)
622. {
623. res[now].pb(g[b].back());
624. g[b].pop\_back();
625. res[now].pb(g[K-b-c].back());
626. g[K-b-c].pop\_back();
627. now++;
628. if(now==cnt)break;
629. }
630. if(now==cnt)break;
631. }
632. if(now==cnt)break;
633. }
634. fo0(i,n)g[s[i]].clear();
635. }
636. }
638. namespace solveTest
639. {
640. std::priority\_queue<pii>q;
641. std::set<pii>p;
642. std::vector<int>g[N];
643. int gc[N];
644. pii t[N];
646. inline ll solve\_old(int K)
647. {
648. fo0(i,n)t[i]=mp(s[i],i);
649. std::sort(t,t+n);
650. int cnt=m/K,rem=K\*cnt;
651. fo0(i,cnt)q.push(mp(K,i));
652. fd0(i,n)
653. {
654. if(!q.empty()&&q.top().xx>=t[i].xx)
655. {
656. pii v=q.top();q.pop();
657. rem-=t[i].xx;
658. if(v.xx-=t[i].xx)q.push(v);
659. }
660. }
661. while(!q.empty())q.pop();
662. *//out,K,' ',rem,'\n';*
663. return(ll)K\*K+(ll)rem\*rem;
664. }
666. inline ll solve(int K)
667. {
668. int now=0,u=0,cnt=m/K;
669. fo0(i,n)gc[s[i]]++;
670. for(int c=0;c<=K;c++)
671. {
672. for(int b=K-c;b>=0;b--)
673. {
674. while(b!=K-b-c?gc[b]&&gc[K-b-c]:gc[b]>1)
675. {
676. gc[b]--;
677. gc[K-b-c]--;
678. u+=c;
679. now++;
680. if(now==cnt)break;
681. }
682. if(now==cnt)break;
683. }
684. if(now==cnt)break;
685. }
686. u+=(cnt-now)\*K;
687. fo0(i,n)gc[s[i]]=0;
688. return(ll)u\*u+(ll)K\*K;
689. }
690. }
692. inline void solve()
693. {
694. if(type==0)solveType0::solve();
695. else if(type==1)solveType1::solve();
696. else if(type==2)solveType2::solve();
697. else if(type==3)solveType3::solve();
698. else if(type==4)solveType4::solve();
699. *//else solve0::solve();*
700. }
702. inline void test(int T)
703. {
704. lf sum=0;
705. int cnt=0;
706. while(1)
707. {
708. gen(T);
709. startTime=clock();
710. solve();
711. ll t=score();
712. if(t!=-1)sum+=sqrt(t),cnt++;
713. printf("sum=%.2lf cnt=%d avg=%.2lf ",sum,cnt,sum/cnt);
714. if(t==-1)printf("Failed ");
715. else printf("score=%.2lf ",sqrt(t));
716. printf("\n");
717. }
718. }
720. inline void draw()
721. {
722. freopen("out.txt","w",stdout);
723. fo0(i,2)gen(4);
724. int step=10,l=1,r=5000;
725. *//4 1000 20000*
726. out,(r-l)/step+1,'\n';
727. for(int i=l;i<=r;i+=step)
728. {
729. printf("%d %.2lf\n",i,sqrt(solveTest::solve(i)));
730. }
731. }
733. int main()
734. {
735. *//test(0);return 0;*
736. *//draw();return 0;*
737. input();
738. *//out,type,'\n';*
739. *//gen(4);*
740. solve();
741. ll t=score();
742. *//printf("%lld %.2lf\n",t,sqrt(t));*
743. output();
744. }

**Language:**C++