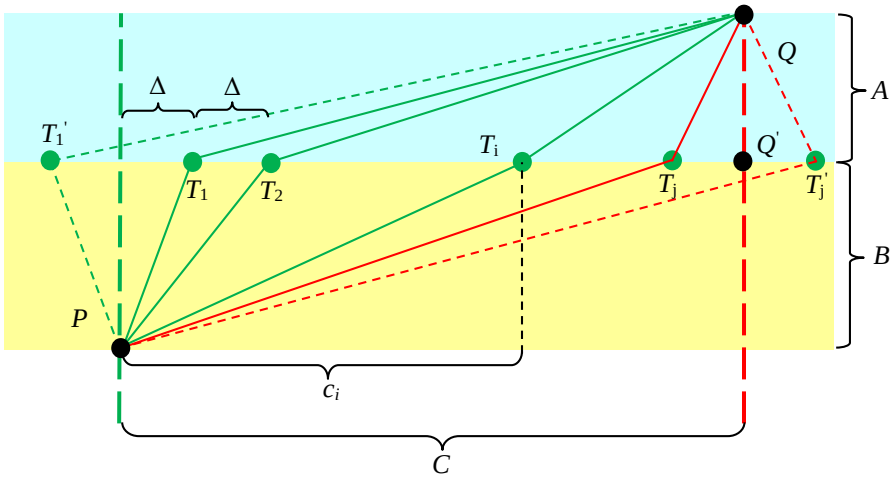
АНАЛИЗ НА РЕШЕНИЕТО НА ЗАДАЧА В#. ПОМОООЩ!

Ако Аялайя все пак успява да се придвижва към брега, то точката на срещата ѝ с Жгмътц ще е някъде по червения перпендикуляр (виж схемата по-долу) между началното ѝ положение Q и пресечката на перпендикуляра с брега Q'. Търсим такова положение на точката на срещата, че до нея двамата герои да достигнат за едно и също време. Ако разработим средство с което да определим времето $t_ж$, за което Жгмътц ще стигне до предполагаема точка на срещата Q_i, можем да сравним t_a – времето на Аялайя за достигане до Q_i, и $t_ж$. Ако $t_ж < t_a$ значи трябва да преместим Q_i към Q, в противен случай – местим я към Q'. Местенето извършваме с постоянно намаляващи стъпка и участък за местене върху $\overline{QQ'}$дотогава, докато участъкът се смали под някаква разумна точност (напр. зададената точност за резултата от задачата). Описаният подход е всъщност двоично търсене на местоположението на точката на срещата върху $\overline{QQ'}$.

Ето и как да разработим средството за определяне на $t_ж$. Обясненията са илюстрирани на схемата и отразяват предположението, че мястото на срещата е в точка Q.

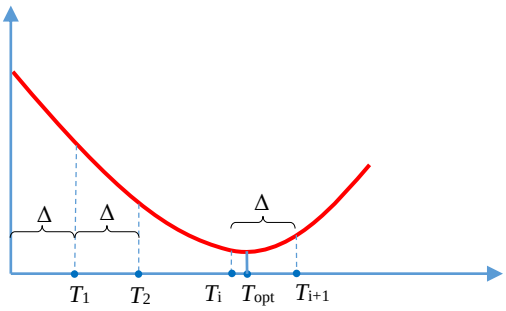
Заради разликите в скоростите на придвижване по пясъка и във водата не е ясно дали отиването до Аялайя по най-краткия път L ще е и най-бързо. Трябва да проверим няколко различни маршрута, за да намерим най-изгодния по време.

Различните маршрути са изградени от по две отсечки – едната от мястото на Жгмътц (точка P на схемата) до точка T_i върху бреговата линия и втората от точката T_i до местоположението на Аялайя (точка Q). Подходът за решаване на задачта е да определим времената за изминаване на различните възможни маршрути и от тях да вземем най-малкото време.



Точката T_{opt}, минаването през която гарантира най-бърз маршрут, задължително се намира между перпендикулярите от местоположенията на героите към бреговата линия. За доказателство на това твърдение да предположим, че оптимумът е в T_1' (на схемата е вляво от зеления перпендикуляр). Точката T_1' е симетрична относно зеления перпендикуляр на точката T_1 , която се намира по бреговата линия между двата перпендикуляра. От симетрията следва, че $\overline{T_1'P} = \overline{T_1P}$ и значи ще се изминат за едно и също време. Обаче разстоянието $\overline{T_1'Q} > \overline{T_1Q}$ и ще бъде изминато за по-дълго време. Т. е. T_1' не може да е оптимум, понеже съществува поне един по-бърз маршрут, преминаващ през точка по бреговата линия, лежаща между двата перпендикуляра. Подобни съображения са валидни и когато набедим за T_{opt}, точка по-вдясно от червения перпендикуляр (вижте напр. T_j' и симетричната ѝ T_j).

За да намерим коя измежду точките T_i е T_{opt}, постъпваме по следния начин. Започвайки от мястото, в което се пресичат зеленият перпендикуляр и бреговата линия, плъзгаме по брега точка с някаква стъпка Δ. За текущото положение на точката изчисляваме времето за преминаване на маршрута, който съдържа точката. Ако времето е по-малко от времето за изминаване на маршрута през предишното положение на точката, значи се движим в правилна посока при търсенето на T_{opt} (вижте схемата с графиката на функцията).Когато за някакво положение на точката времето за изминаване на маршрута е по-голямо от времето за изминаване на маршрута през предишното ѝ положение, значи сме подминали T_{opt} В този случай намаляваме стъпката Δ наполовина и обръщаме посоката на плъзгане на точката. Така местоположението на плъзганата точка се колебае около T_{opt} с постояннно намаляваща амплитуда докато в някакъв момент достигнем до положение, практически съвпадащо с T_{opt}.



Оформяме изложения алгоритъм като програмен код и тъкмо да го запратим към тестващата система се сещаме, че пред Жгмътц има и допълнителни възможности за минимизиране на времето за достигане до Аялайя – може да пробяга известно разстояние между зеления и червения перпендикуляри по ръба на асфалтовата алея, където скоростта му на придвижване е най-голяма! И тук възниква въпросът „Колко да е това известно растояние?“ За да отговорим, постъпваме по начин, подобен на описания при търсене на минималното време за прекосяване на пясъчната ивица и водното разстояние между героите. Няма да преразказваме отново алгоритъма, може да се прочете в приложения по долу програмен код.

Сорс-кодът на програмата

```
#include <iostream>
#include <cmath>
#include <iomanip>
#define PRECISION (1.0e-5)
using namespace std;

int L,A,B;
double Va, Vp, Vv, V;

double get_min_time_through_sand_and_water (double k1, double k2_1=A, double k2_2=B) {
  double step, sprev, b, s;
  for (step=b=k1/2, sprev=k2_2/Vp+hypot(k1,k2_1)/Vv; true; sprev=s, b+=step) {
    s = hypot(b,k2_2)/Vp + hypot(k1-b,k2_1)/Vv;
    if (fabs(s-sprev)<= PRECISION) return s;
    if (s > sprev) step = -step/2;
  }
}

double get_min_time (double k1, double k2_1, double k2_2=B){
  double s, sprev, step, c;
  sprev=get_min_time_through_sand_and_water(k1,k2_1,k2_2);
  for (c=step=k1/2 ; true; c+=step, sprev=s) {
    s = fabs(c)/Va + get_min_time_through_sand_and_water(fabs(k1-c),k2_1,k2_2);
    if (fabs(s-sprev) <= PRECISION) return s;
    if (s > sprev) step = -step / 2;
  }
}

int main() {

  cin >> L >> A >> B >> Va >> Vp >> Vv >> V;
  double k1 = sqrt(L*L-(A+B)*(A+B)), tj,a,b,e,t,dt;
```

```
  if (V==0.0) tj = get_min_time(k1,A);
  else for (b=0,e=A;e-b>PRECISION;(tj<t?e:b)=a) {
      a = (b+e)/2;
      t = a/V;
      tj = get_min_time(k1,A-a);
      dt = fabs(t-tj);
      if (dt<PRECISION) break;
  }

  cout << fixed << setprecision(3) << tj << endl;
  return 0;
}
```

Автор: Евгений Василев