**210711 牛鹏军 21374389**

题目描述如下：

|  |
| --- |
| dfed98b2ee00149f77ad97a0d10aa3b |

算法实现如下：

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include <math.h>  #include <time.h>  #define A 5.0  #define B 3.0  #define EPSILON 1e-8 // 误差限  // 计算函数 f(t, x, y) 的值  double f(double t, double x, double y) {  return (A \* cos(t) - x) \* (A \* cos(t) - x) + (B \* sin(t) - y) \* (B \* sin(t) - y);  }  // 计算函数 f(t, x, y) 对 t 的导数  double df(double t, double x, double y) {  return 2 \* (A \* cos(t) - x) \* (-sin(t)) + 2 \* (B \* sin(t) - y) \* B \* cos(t);  }  // 使用牛顿法寻找函数 f(t, x, y) 的最小值  double newton\_method(double x, double y) {  double t = atan2(y, x); // 初始化角度为 0  double Y, d, old, lambda;  int i = 0, j = 0;  for (i = 0; i < 64; i++) {  old = f(t, x, y);  d = df(t, x, y);  if (fabs(d) < 1e-100)  return 0;  d = old / d; // 牛顿法中的步长  lambda = 1.0;  // 二次插值  for (j = 0; j < 8; j++) {  Y = f(t - lambda \* d, x, y);  if (fabs(old) > fabs(Y))  break;  lambda \*= (-0.5); // 缩小步长  }  if (j < 8) {  t -= lambda \* d;  } else {  t -= d; // 使用步长更新角度  Y = f(t, x, y);  }  return fabs(Y); // 返回函数值的绝对值  }    return 0;  }  int main() {  int i;  time\_t t;  srand((unsigned)time(&t));  for (i = 0; i < 20; i++) {  // 生成随机的 (x, y) 坐标  double x = (double)rand() / RAND\_MAX \* 10.0 - 5.0; // 生成 -5 到 5 之间的随机数  double y = (double)rand() / RAND\_MAX \* 10.0 - 5.0;  // 使用牛顿法计算最小值  double t\_min = newton\_method(x, y);  // 打印结果  printf("x=%7.4f y=%7.4f distance=%7.4f\n", x, y, sqrt(f(t\_min, x, y)));  }  return 0;  } |

输出结果如下：

|  |
| --- |
|  |
| x= 4.8053 y= 3.6187 distance=10.2827  x=-4.4790 y=-4.1266 distance= 5.6161  x=-1.6289 y= 0.0133 distance= 2.9604  x= 3.2815 y=-2.9095 distance= 4.1996  x=-4.7711 y=-4.0997 distance= 8.3847  x= 2.2735 y= 2.0495 distance= 2.3253  x=-1.1364 y=-3.4872 distance= 7.3707  x= 4.7324 y=-4.7589 distance=10.2066  x=-2.7874 y=-1.1937 distance= 5.0227  x= 2.4920 y=-1.9369 distance= 3.9337  x= 0.3859 y= 4.5453 distance= 4.7948  x=-1.0262 y=-0.8968 distance= 4.8016  x= 1.0625 y=-4.6365 distance= 7.9897  x=-4.7421 y= 2.1773 distance= 8.7888  x= 1.6274 y=-2.3907 distance= 4.4480  x=-3.2028 y= 2.4349 distance= 8.5407  x=-2.8951 y=-0.2126 distance= 2.0257  x= 2.3406 y=-4.1571 distance= 8.3277  x=-2.0403 y= 0.4122 distance= 6.9863  x= 1.6228 y= 2.3601 distance= 3.6570 |