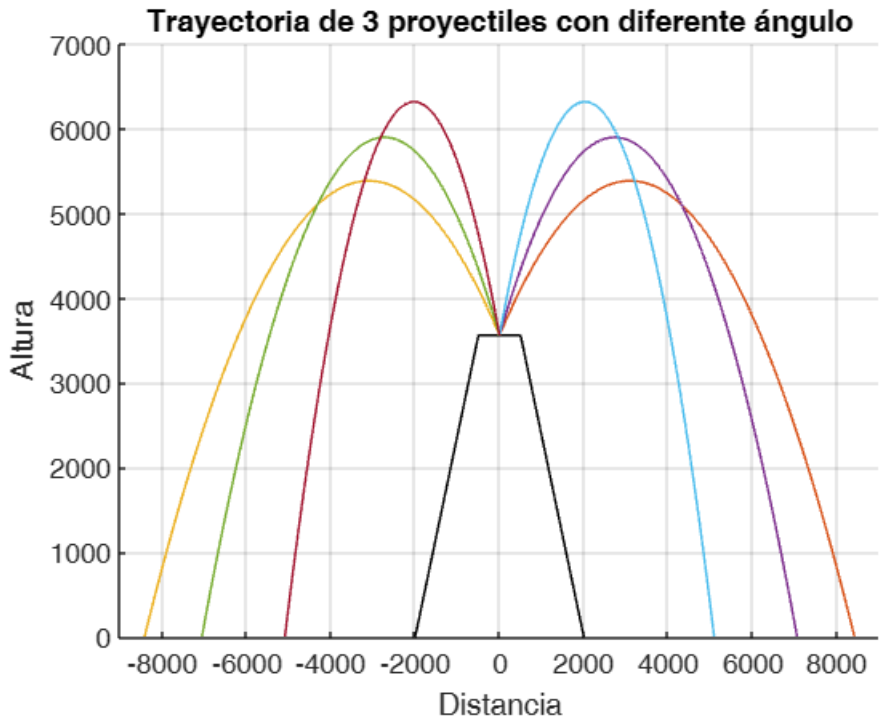
```
clc;clf;clear all;

figure(1);
hold on;

% Pide los valores de entrada al usuario: Altura del volcán, masa del
% proyectil, velocidad inicial y densidad del aire

vol = input('Altura del volcán (m):');
masa = input('Masa (kg):');
vi = input('Velocidad inicial (m/s):');
d = input('Densidad del aire:');

% Crea el volcán con una base de 4000 (2000de cada lado), con una pendiente
% de 1000/500. Al final usa la función plot para dibujar el volcán.

b = 2000;
ba = 500;
i = [-b, -ba, ba, b];
j = [0, vol, vol, 0];
plot(i, j, 'k')

% La gráfica muestra los valores de x entre -9000 y 9000, y los valores en
% y de 0 a 7000. Grid on se utiliza para que salga con cuadrícula.

xlim([-9000, 9000])
ylim([0 7000])
xlabel('Distancia horizontal')
ylabel('Altura')
grid on

% Este ciclo se repite 3 veces para crear las trayectorias de 3 proyectiles
% con diferente ángulo, pidiendo un ángulo de salida cada que se haga el
% ciclo y llamandi a la función dibujar cada vez.

for j = 1:3

    ang = input('Ángulo de salida: ');
    deg = ang;
    dibujar(deg, vol, vi, masa, delta, d);


end
title('Trayectoria de 3 proyectiles con diferente ángulo')
xlabel('Distancia');
ylabel('Altura');
```




```
% esta función recibe como parámetros los grados, la altura del volcán, la
% velocidad inicial, la masa, la diferencia de tiempo y la densidad del
% volcán.
```

```
function dibujar(deg, vol, vi, masa, delta, d)

    y0 = vol;
    v0 = vi;
    delta = 0.5;

    % Drag coefficient = el coeficiente de fricción de un objeto esférico
    % con el aire
    cd = 0.45;
    % Área frontal del proyectil
    a = 0.785;

    g = 9.8;
    x0 = 0;
    % Velocidades iniciales en cada componente
    vx0 = v0*cos(deg2rad(deg));
    vy0 = v0*sin(deg2rad(deg));
    % Resistencia del aire
    k = (cd * d * a) / 2;

    i = 1;
```

```
x(i) = x0;
vx(i) = vx0;

% De la fórmula de la Segunda Ley de Newton F = ma, despejamos a para
% conseguir la aceleración en x.
ax(i) = (-k/masa);

y(i) = y0;
vy(i) = vy0;
% La aceleración en y menos la aceleración de la gravedad
ay(i) = -g - (k/masa);
v(i) = v0;
t(i) = 0;

% Se repetira el ciclo hasta que y sea el piso (0)
while min(y) > 0

    % El método de verlet encuentra la pendiente entre dos puntos
    % cercanos para representar una aproximación a la gráfica real. Los
    % dos puntos a evaluar son (x1,y1) y (x2,x2) obteniendo la
    % pendiente entre ambos con el cambio de velocidad entre ambos
    % puntos y también usando su aceleración. Los pasos están marcados
    % por delta (la diferencia del tiempo)

    % Tiempo en cada paso = tiempo anterior más la diferencia del
    % tiempo
    t = [t, t(i) + delta];

    % Las velocidades en el paso se consigue utilizando la ecuación
    % Vf=Vi+at, cambiando la i en cada paso
    vx = [vx, vx(i) + (ax(i) * delta)];
    vy = [vy, vy(i) + (ay(i) * delta)];

    % Como las velocidades se actualizan antes de estos pasos, i+1 ya
    % existe en los datos de velocidades para el paso i en x y y.
    % La velocidad inicial se le suma a la velocidad del siguiente paso
    % multiplicado por la diferencia del tiempo para encontrar su
    % desplazamiento en el intervalo de delta
    x = [x, x(i) + (vx(i + 1) * delta)];
    y = [y, y(i) + (vy(i + 1) * delta)];

    v = [v, sqrt(vx(i + 1)^2 + vy(i + 1)^2)];

    % Se actualiza la aceleración
    ax = [ax, (-k/masa)];
    ay = [ay, -g - (k/masa)];

    % Se le suma un paso cada vez que se haga un punto
    i = i + 1;
```

```
    end
    % Dibuja la trayectoria con los valores de x y y del lado derecho e
    % izquierdo
    plot(x, y)
    plot(-x, y)
end
```