1. **Quantum Wave Packets and Time Evolution**
   In this exercise we are going to solve the time-dependent Schrödinger equation

$$i\frac{\partial}{\partial t}\psi(x,t) = -\frac{1}{2m}\frac{\partial^2}{\partial x^2}\psi(x,t) + V(x)\psi(x,t) \tag{1}$$

   numerically for $m = 1$. We consider a wave packet of the form

$$\psi(x,t_0) = \exp\left(-\frac{1}{2}\left(\frac{x-x_0}{\sigma_0}\right)^2\right)\exp(ik_0x), \tag{2}$$

   which is centered around x_0 and multiplied by a plane wave with momentum k_0.

   (a) Rewrite the wave function and the Schrödinger equation in terms of two real functions.

   (b) Rewrite your result to discretized time and space using finite differences.

   (c) In order to improve the conservation of probability you have to evaluate the real and imaginary part of your derived recursion formula at different times.

   (d) Implement the staggered-leapfrog algorithm to solve the 1D time-dependent Schrödinger equation for no potential as well as for infinitely high walls, the harmonic potential, and a step potential. Your initial wave function should be of the form given above. Try $\sigma_0 = 0.5$ and $k_0 = 17\pi$ or $k_0 = 0$. Although $\Delta t = (\Delta x/2)^2$ should work rather nicely, study the stability for smaller/larger Δt.

   (e) Generate a ρ over $x - t$ plot for each of your solutions. How does the staggered-leapfrog algorithm affect the formula of the probability density ρ?

   (f) What physical properties/processes do you observe?

   (g) In the lecture, it was demonstrated that instead of using staggered times, the conservation of probability can also be achieved by using the wave function "two steps in the past" (instead of one) in the recursion. Implement this algorithm and compare it to the staggered-leapfrog algorithm.

   (h) Generalize your program to 2D and study the behavior of a 2D wave packet in at least one of the above non-trivial potentials.

   (i) Generate ρ over $x - y$ plots for different times. *Optional: You can generate lots of such plots and combine them into a short video, e.g. using* `mencoder`.