```julia
using PyPlot;

u0(x) = if abs(x) <= 0.5;
  (cos(pi * x))^2;
else
  0;
end;

const x0 = -1.0;
const x1 = 3.0;

const t0 = 0.0;
const t1 = 2.4;

const lambda = 0.8;

for h in [1./10.; 1./20.; 1./40.]

  println("Forward-Time Backward-Space, h = $h");

  xs = collect(x0:h:(x1+eps()));
  k = h * lambda;
  ts = collect(t0:k:(t1+eps()));
  us = zeros(length(xs), length(ts));
  us[:, 1] = map(u0, xs); # initial conditions

  for n in 1:length(ts)-1, m in 2:length(xs)
    us[m, n+1] = us[m, n] - lambda * (us[m, n] - us[m-1, n]);
  end

  plot(xs, us[:, 1], "k-"; label=@sprintf("\$t = %.3f\$", ts[1]));
  plot(xs, us[:, div(length(ts), 2)], "k--"; label=@sprintf("\$t = %.3f\$",
                                                     ts[div(length(ts), 2)]));
  plot(xs, us[:, end], "k-."; label=@sprintf("\$t = %.3f\$", ts[end]));
  legend(; loc=3);
  title("Forward-Time Backward-Space, \$h = $h\$");
  xlabel("x");
  ylabel("u");
  ylim([0.0; 1.2]);
  savefig("forward-time_backward-space_h-$h.png");
  clf();
end

for h in [1./10.; 1./20.; 1./40.]

  println("Forward-Time Central-Space, h = $h");

  xs = collect(x0:h:x1);
  k = h * lambda;
  ts = collect(t0:k:t1);
  us = zeros(length(xs), length(ts));
  us[:, 1] = map(u0, xs); # initial conditions

  for n in 1:length(ts)-1
    for m in 2:length(xs)-1
      us[m, n+1] = us[m, n] - lambda * (us[m, n] - us[m-1, n]);
    end
    us[end, n+1] = us[end-1, n+1];
  end

  plot(xs, us[:, 1], "k-"; label=@sprintf("\$t = %.3f\$", ts[1]));
  plot(xs, us[:, div(length(ts), 2)], "k--"; label=@sprintf("\$t = %.3f\$",
                                                     ts[div(length(ts), 2)]));
  plot(xs, us[:, end], "k-."; label=@sprintf("\$t = %.3f\$", ts[end]));
  legend(; loc=3);
  title("Forward-Time Central-Space, \$h = $h\$");
  xlabel("x");
  ylabel("u");
  ylim([0.0; 1.2]);
  savefig("forward-time_central-space_h-$h.png");
  clf();
end

for h in [1./10.; 1./20.; 1./40.]

  println("Lax-Friedrichs, h = $h");

  xs = collect(x0:h:x1);
  k = h * lambda;
  ts = collect(t0:k:t1);
  us = zeros(length(xs), length(ts));
  us[:, 1] = map(u0, xs); # initial conditions

  for n in 1:length(ts)-1
    for m in 2:length(xs)-1
      us[m, n+1] = (0.5 * (us[m+1, n] + us[m-1, n]) -
                    0.5 * lambda * (us[m+1, n] - us[m-1, n]));
    end
    us[end, n+1] = us[end-1, n+1];
  end

  plot(xs, us[:, 1], "k-"; label=@sprintf("\$t = %.3f\$", ts[1]));
  plot(xs, us[:, div(length(ts), 2)], "k--"; label=@sprintf("\$t = %.3f\$",
                                                     ts[div(length(ts), 2)]));
  plot(xs, us[:, end], "k-."; label=@sprintf("\$t = %.3f\$", ts[end]));
  legend(; loc=3);
  title("Lax-Friedrichs, \$h = $h\$");
  xlabel("x");
  ylabel("u");
  ylim([0.0; 1.2]);
  savefig("lax-friedrichs_h-$h.png");
  clf();
end

for h in [1./10.; 1./20.; 1./40.]

  println("Leapfrog, h = $h");

  xs = collect(x0:h:x1);
  k = h * lambda;
  ts = collect(t0:k:t1);
  us = zeros(length(xs), length(ts));
  us[:, 1] = map(u0, xs); # initial conditions

  # one step of FTCS
  for m in 2:length(xs)-1
    us[m, 2] = us[m, 1] - lambda * (us[m, 1] - us[m-1, 1]);
  end
  us[end, 2] = us[end-1, 2];

  for n in 2:length(ts)-1
    for m in 2:length(xs)-1
      us[m, n+1] = us[m, n-1] - lambda * (us[m+1, n] - us[m-1, n]);
    end
    us[end, n+1] = us[end-1, n+1];
  end

  plot(xs, us[:, 1], "k-"; label=@sprintf("\$t = %.3f\$", ts[1]));
  plot(xs, us[:, div(length(ts), 2)], "k--"; label=@sprintf("\$t = %.3f\$",
                                                     ts[div(length(ts), 2)]));
  plot(xs, us[:, end], "k-."; label=@sprintf("\$t = %.3f\$", ts[end]));
  legend(; loc=3);
  title("Leapfrog, \$h = $h\$");
  xlabel("x");
  ylabel("u");
  ylim([0.0; 1.2]);
  savefig("leapfrog_h-$h.png");
  clf();
end
```

```
function total_variation(us::Vector{Float64})
  sum = 0.0;
  for j=2:length(us)-1
    sum += abs(us[j+1] - us[j]);
  end
  return sum;
end

all_vars = Dict();
for h in [1./5.; 1./10.; 1./100.]

  println("Lax-Friedrichs, h = $h");

  xs = collect(-2.0:h:(15.0+0.8*h));
  k = h * lambda;
  ts = collect(0.0:k:(10.0+0.8*k));
  us = zeros(length(xs), length(ts));
  us[:, 1] = map(x -> (abs(x) <= 1) ? 2.0 : 0.0, xs); # initial conditions
  vars = zeros(length(ts));
  vars[1] = total_variation(us[:, 1]);

  for n in 1:length(ts)-1
    for m in 2:length(xs)-1
      us[m, n+1] = (0.5 * (us[m+1, n] + us[m-1, n]) -
                    0.5 * lambda * (us[m+1, n] - us[m-1, n]));
    end
    us[end, n+1] = us[end-1, n+1];
    println("t = $(ts[n+1]), TV = $(total_variation(us[:, n+1]))");
    vars[n+1] = total_variation(us[:, n+1]);
  end

  @show sum(vars) / length(vars);
  plot(ts, vars);
  title("Lax-Friedrichs, \$h = $h\$");
  xlabel("Time");
  ylabel("Total Variation");
  savefig("lax-friedrichs-TV_h-$h.png");
  clf();

  plot(xs, us[:, 1], "k-"; label=@sprintf("\$t = %.3f\$", ts[1]));
  plot(xs, us[:, div(length(ts), 2)], "k--"; label=@sprintf("\$t = %.3f\$",
                                                     ts[div(length(ts), 2)]));
  plot(xs, us[:, end], "k-."; label=@sprintf("\$t = %.3f\$", ts[end]));
  legend(; loc=3);
  title("Lax-Friedrichs, \$h = $h\$");
  ylim([0.0; 2.5]);
  savefig("lax-friedrichs-c2_h-$h.png");
  clf();

  all_vars[h] = (copy(ts), copy(vars));
end

linetypes = ["k-", "k--", "k-."];
for ((h, data), linetype) in zip(all_vars, linetypes)
  plot(data[1], data[2], linetype; label=@sprintf("\$h = %.3f\$", h));
end
legend(; loc=3);
xlabel("Time");
ylabel("Total Variation");
savefig("lax-friedrichs-TV_all.png");
```