```
function [res] = montecarlo2(n,t,c)
    % Evaluates P[X>t] where X follows a normal distribution
    % with variance reduction
    x = randn(1,n)+c;
    res = exp(−c^2/2)*sum((x>t).*exp(c*(c−x)))/n;

function [res] = montecarlo(n,t)
    % Evaluates P[X>t] where X follows a normal distribution
    x = randn(1,n);
    res = sum(x>t)/n;
function [b]=piece()
    % piece provides a sample of a Bernoulli(0.5) distribution
    b = (rand()<0.5)*1;

function [u,k] = rbern(p)
    % This function simulates a Bernoulli random variable with parameter p
    k = 1;
    bk = piece();
    pk = p − floor(p);
    while ((bk<0.5)==(pk<0.5))
        bk = piece();
        pk = pk*2 − floor(pk*2);
        k = k + 1;
    end
    if (bk == 0)
        u = 1;
    else
        u = 0;
    end
function [x] = rcauchy()
    % Simulates samples from a Cauchy random variable
    x = tan(pi*(rand−0.5));

function [y] = rellipse(a,b,n)
    % Simulates n samples from a uniform distribution on an ellipse
    % (x/a)^2 + (y/b)^2 = 1
    x1 = 2*(rand(1,2*n)−0.5);
    x2 = 2*(rand(1,2*n)−0.5);
    test = (x1.*x1 + x2.*x2);
    masque = find(test < 1);
    y = [x1(masque)*a;x2(masque)*b];
    y = y(1:2,1:n);

function [x] = rgauss(m,s)
    % Simulates a gaussian random variable by inversion
    x0 = − sqrt(2) * erfcinv(2*rand());
    x = (x0 * s + m);

function [y] = rproba(s,p)
    % Simulates a discrete random variable with law p on support s
    cdf = cumsum(p);
    u = rand();
    x = 1;
%    while (cdf(x) < u)
%        x = x + 1;
%    end

% on peut aussi écrire sans boucle
x = sum(cdf < u) + 1;
y = s(x);
function [u] = runif(e)
    % This function simulates a uniform va on [0,1] based on piece.m
    % e indicates maximum precision
    u = 0;
    k = 0;
    while (e < 1 / 2^k)
        k = k+1;
        u = u + piece() / 2^k;
    end
```



```
% Prépa Agreg 2007 ENS Cachan
% Correction du TP1 Statistiques (Matlab 7.4.0 + Stixbox)
% Li−Thiao−Té Sébastien

% 1 Lancers de pièces

% Vérification de piece
n = 10000;
echantillon = zeros(n,1);
for k = 1:n
    echantillon(k) = piece();
end
length(find(echantillon == 0))/n % Environ 0.5

% Vérification de runif
n = 100000;
echantillon = zeros(n,1);
for k = 1:n
    echantillon(k) = runif(0.0001);
end
hist(echantillon,250)

% Vérification de rbern
n = 10000;
p = 0.3;
echantillon = zeros(n,1);
for k = 1:n
    echantillon(k) = rbern(p);
end
length(find(echantillon == 1))/n % Environ p

% Nombre de tirages
n = 10000;
echantillon = zeros(n,1);
for k = 1:n
    [a, nb] = rbern(p);
    echantillon(k) = nb;
end
sum(echantillon) / n % Environ 2

% Convergence des sommes partielles
x = ones(1,10);
cumsum(cumsum(x) .* cumprod(x/2))

%%%%%
% 2 Inversion de la fonction de répartition

% Vérification de rproba
n = 10000;
p = [0.4 0.2 0.3 0.1];
s = [0   3   2   4];
echantillon = zeros(n,1);
for k = 1:n
    echantillon(k) = rproba(s,p);
end
hist(echantillon)

% Vérification de rgauss
m = 0.45;
s = 1.1;
n = 10000;
echantillon = zeros(n,1);
for k = 1:n
    echantillon(k) = rgauss(m,s);
end
mean(echantillon)
std(echantillon)
qqnorm(echantillon)
```

```
% Vérification de rcauchy
n = 10000;
echantillon = zeros(n,1);
for k = 1:n
    echantillon(k) = rcauchy();
end
pas = 0.5;
edges = −10:pas:10;
histo = histc(echantillon,edges);
histo = histo / sum(histo) / pas;
bar(edges,histo,'histc');
hold on;
x = (−10:pas:10) + pas / 2;
plot(x,1./(pi * (1 + x.^2)));
hold off;

%%%%%
% 3 Méthode de rejet

% Vérification de rellipse
echantillon = rellipse(2,1,10000);
plot(echantillon(1,:),echantillon(2,:),'.');
masque = find(echantillon(1,:)>−1 & echantillon(1,:)<1 & echantillon(2,:)>−0.4 &
 echantillon(2,:)<0.3);
length(masque)/10000 % environ 2*0.7/3.14/2/1

%%%%%
% 4 Méthode de Monte Carlo

% Méthode de Monte Carlo simple
n = 100;
t =qnorm(0.99);
montecarlo(n,t)
1−pnorm(t)

n = 1000;
echantillon = zeros(n,1);
for k = 1:n
    echantillon(k) = montecarlo(100,t);
end
hist(echantillon);
mean(echantillon)
std(echantillon)

% Méthode de Monte Carlo avec translation
n = 100;
c = sqrt(t^2+2*log(t));
montecarlo2(n,t,c)
1−pnorm(t)

n = 1000;
echantillon2 = zeros(n,1);
for k = 1:n
    echantillon2(k) = montecarlo2(100,t,c);
end
hist(echantillon2);
mean(echantillon2)
std(echantillon2)

% Comparaison des deux méthodes
pas = 0.001;
edges = 0:pas:0.02;
histo = histc(echantillon,edges);
histo1 = histo / sum(histo) / pas;
histo = histc(echantillon2,edges);
histo2 = histo / sum(histo) / pas;
plot(edges,histo1,'b');
hold on;
plot(edges,histo2,'g');
```



```
hold off;
```