Consider a single serve queue where customers arrive according to a Poisson process with rate 2 per minute and the service times are exponentially distributed with mean 1 minute. Let T_i denote the amount of time that customer i spends in the system. We are interested in using simulation to estimate $\theta = E[T_1 + \cdots + T_{10}]$.

```julia
import Statistics: mean, var
function Run(arrival_t, service_t, n)
    depart_t = Inf
    N = 0
    i = 1
    new = 1
    spend = Array{Float64}(undef, n)
    while i in 1:n
        if min(depart_t, arrival_t[new]) == depart_t
            spend[i] = depart_t - arrival_t[i]
            N -= 1
            i += 1
            if N == 0
                depart_t = Inf
            else
                depart_t += service_t[i]
            end
        else
            if N == 0
                depart_t = arrival_t[new] + service_t[i]
            end
            N += 1
            new += 1
        end
    end
    return sum(spend)
end
```

Run 函數是在給定客人數量、到達時間與每位客人所需的服務時間的參數後，計算客人們在系統內時間的總和。

變數名稱分別代表，

depart_t：現在正在被服務的客人的離開時間

N：系統總人數

i：將準備要計算花費時間的客人的 index

new：下一位將要進門的客人的 index

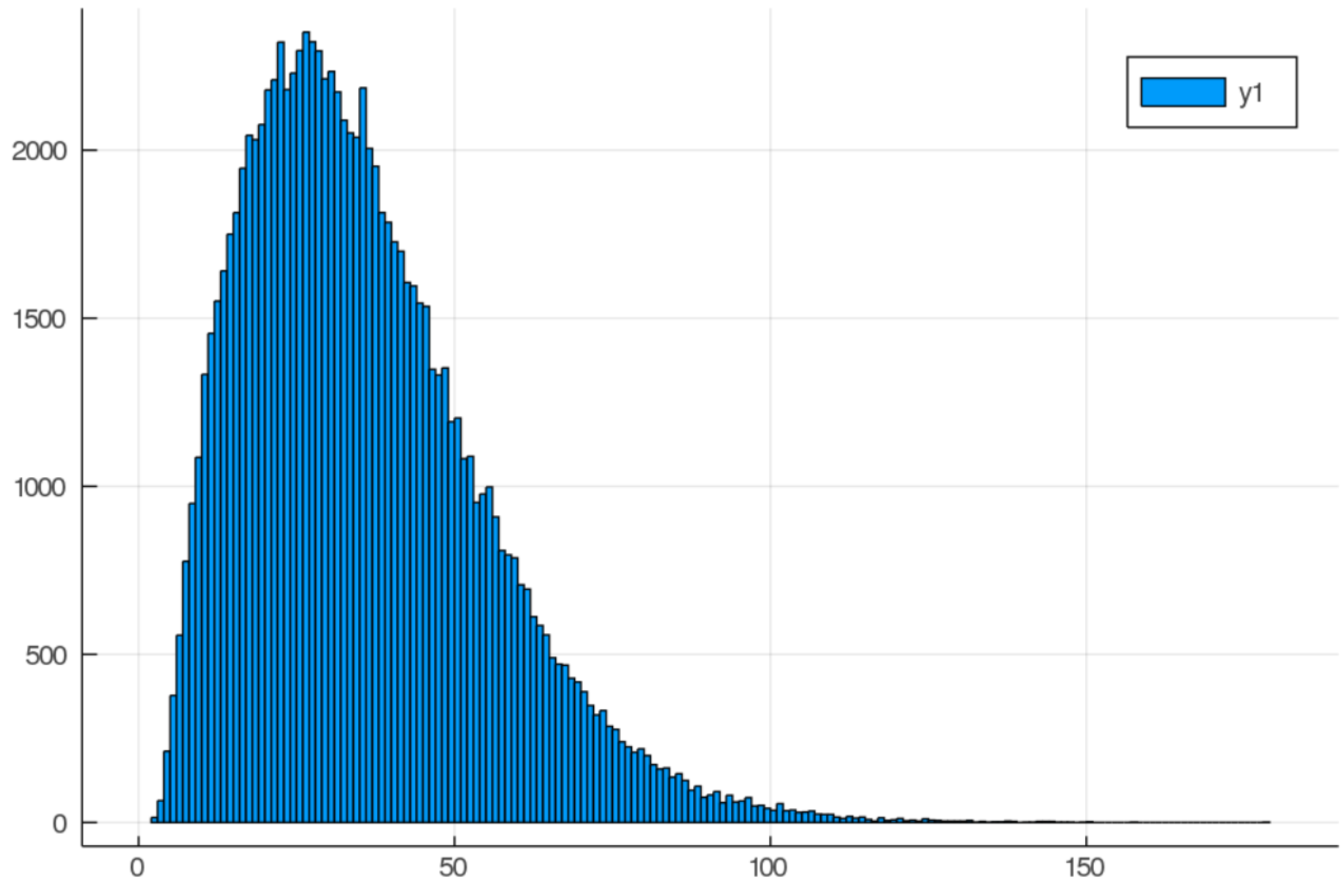
spend：紀錄每個客人花費在系統內的時間

(a) Do a simulation to estimate the variance of the raw simulation estimator. That is, estimate $\mathrm{Var}(T_1 + \cdots + T_{10})$.

```julia
# (a)
function f1(n::Int, arrival_rate, service_rate)
    arrival_t = cumsum(-log.(rand(n)) / arrival_rate)
    push!(arrival_t, Inf)
    service_t = -log.(rand(n)) / service_rate
    return Run(arrival_t, service_t, n)
end   > f1

using Plots   ✓
a1 = [f1(10, 2, 1) for _ in 1:100000]   > Vector{Float64} with 100000 elements

mean(a1)   35.70107943212388
variance_1 = var(a1)   360.064293633708
histogram(a1)   Plot{Plots.GRBackend() n=1}
```



平均值 35.70107943212388
變異數 360.064293633708

(b) Do a simulation to determine the improvement over the raw estimator obtained by using antithetic variables.

```
# (b)
function f2(n::Int, arrival_rate, service_rate)
    u = rand(n)
    arrival_t1 = cumsum(-log.(u) / arrival_rate)
    arrival_t2 = cumsum(-log.(1 .- u) / arrival_rate)
    push!(arrival_t1, Inf)
    push!(arrival_t2, Inf)
    u = rand(n)
    service_t1 = -log.(u) / service_rate
    service_t2 = -log.(1 .- u) / service_rate
    return (Run(arrival_t1, service_t1, n) + Run(arrival_t2, service_t2, n))/2
end    > f2

a2 = [f2(10, 2, 1) for _ in 1:100000]    > Vector{Float64} with 100000 elements
mean(a2)    35.740105535594104
var(a2) / variance_1    0.2228542710354517
```

平均值 35.740105535594104

變異數是第一題的 22%

(c) Do a simulation to determine the improvement over the raw estimator obtained by using $\sum_{i=1}^{10} S_i$ as a control variate, where S_i is the ith service time.

```
# (c)
import LinearAlgebra.dot    ✓

function f3(n::Int, arrival_rate, service_rate)
    arrival_t = cumsum(-log.(rand(n)) / arrival_rate)
    push!(arrival_t, Inf)
    service_t = -log.(rand(n)) / service_rate
    return [Run(arrival_t, service_t, n), sum(service_t)]
end    > f3

nsim = 100000    100000
x = Array{Float64}(undef, nsim)    > Vector{Float64} with 100000 elements
y = Array{Float64}(undef, nsim)    > Vector{Float64} with 100000 elements

for i in 1:nsim, (a, b) in tuple(f3(10, 2, 1))
    x[i], y[i] = a, b
end    ✓
c = -dot(x .- mean(x), y .- mean(y)) / dot(y .- mean(y), y .- mean(y))    -5.09…
q3 = x + c * (y .- 10)    > Vector{Float64} with 100000 elements
mean(q3)    35.703646494466916
var(q3) / variance_1    0.27608980206347
```

平均值 35.703646494466916

變異數是第一題的 27%

(d) Do a simulation to determine the improvement over the raw estimator obtained by using $\sum_{i=1}^{10} S_i - \sum_{i=1}^{9} I_i$ as a control variate, where I_i is the time between the ith and $(i+1)$st arrival.

```
# (d)
function f4(n::Int, arrival_rate, service_rate)
    arrival_t = cumsum(-log.(rand(n)) / arrival_rate)
    push!(arrival_t, Inf)
    service_t = -log.(rand(n)) / service_rate
    return [Run(arrival_t, service_t, n), sum(service_t) - sum(diff(arrival_t[1:length(arrival_t)-1]))]
end    > f4

nsim = 100000    100000
x = Array{Float64}(undef, nsim)    > Vector{Float64} with 100000 elements
y = Array{Float64}(undef, nsim)    > Vector{Float64} with 100000 elements

for i in 1:nsim, (a, b) in tuple(f4(10, 2, 1))
    x[i], y[i] = a, b
end    ✓
c = -dot(x .- mean(x), y .- mean(y)) / dot(y .- mean(y), y .- mean(y))    -4.77…
q4 = x + c * (y .- (10 - 9/2))    > Vector{Float64} with 100000 elements
mean(q4)    35.706837393007696
var(q4) / variance_1    0.22105019481225696
```

平均值 35.706837393007696

變異數是第一題的 22%

(e) Do a simulation to determine the improvement over the raw estimator obtained by using the estimator $\sum_{i=1}^{10} E[T_i|N_i]$, where N_i is the number in the system when customer i arrives (and so $N_1 = 0$).

```
# (e)
function Run2(arrival_t, service_t, n)
    depart_t = Inf
    N = 0
    i = 1
    new = 1
    output = Array{Float64}(undef, n)
    while i in 1:n
        if min(depart_t, arrival_t[new]) == depart_t
            i += 1
            N -= 1
            if N == 0
                depart_t = Inf
            else
                depart_t += service_t[i]
            end
```

```
        else
            if N == 0
                depart_t = arrival_t[new] + service_t[i]
            end
            N += 1
            output[new] = N
            new += 1
        end
    end
    return sum(output)
end    > Run2
```

```
function f5(n::Int, arrival_rate, service_rate)
    arrival_t = cumsum(-log.(rand(n)) / arrival_rate)
    push!(arrival_t, Inf)
    service_t = -log.(rand(n)) / service_rate
    return Run2(arrival_t, service_t, n) / service_rate
end    > f5

a5 = [f5(10, 2, 1) for _ in 1:100000]    > Vector{Float64} with 100000 elements
mean(a5)    35.74327
var(a5) / variance_1    0.2876049002931684
```

平均值為 35.74327

變異數是第一題的 28%