Questão 3.

a) $\int_0^1 (1-x^2)^{3/2} dx$

Solução:

Solução da integral:

Fazendo a transformação $u = x^2$, chegaremos a seguinte integral:

$$\frac{1}{2}\int_0^1 (1-u)^{3/2} u^{-1/2} du = \frac{1}{2} B(5/2, 1/2) = \frac{1}{2}\frac{\Gamma(5/2)\Gamma(1/2)}{\Gamma(3)} = \frac{3\pi}{16} = 0.5890486,$$

em que $\Gamma(1/2) = \sqrt{\pi}$, $\Gamma(5/2) = \frac{3\sqrt{\pi}}{4}$ e $\Gamma(3) = 2$.

Método de Monte Carlo:

$$\int_0^1 (1-x^2)^{3/2} dx = E((1-X^2)^{3/2}) \approx \frac{1}{n}\sum_{i=1}^{n} (1-X_i^2)^{3/2}$$

em que $X \sim U(0,1)$.

Código R:

```
n <- 10^7
X <- runif(n)
mean(((1-X^2))^(3/2))
[1] 0.5892025
```

Comparando com o valor exato, temos que é uma boa aproximação até a terceira casa decimal.

b) $\int_{-2}^{2} e^{x+x^2} dx$

Solução: No item b) devemos aplicar uma transformação de variável de modo que o espaço de integração seja o espaço (0,1). Fazendo a transformação $u = \frac{x+2}{4}$ chegaremos a seguinte integral,

$$\begin{aligned}\int_{-2}^{2} e^{x+x^2} dx = 4\int_0^1 exp\big(4u - 2 + (4u-2)^2\big) du &= 4E\big(exp\big(4U - 2 + (4U-2)^2\big)\big) \\ &\approx \frac{4}{n}\sum_{i=1}^{n} exp\big(4U_i - 2 + (4U_i - 2)^2\big)\end{aligned}$$

em que $U \sim U(0,1)$.

Código R:

```
n <- 10^7
U <- runif(n)
4*mean(exp(4*U - 2 +(4*U - 2)^2 ))
[1] 93.11628
```

c) $\int_0^1 \int_0^1 exp(x+y)^2 dxdy$

Solução

No item c) podemos ver a integral como sendo

$$\int_0^1 \int_0^1 exp(x+y)^2 dxdy = E_{XY}(exp(X+Y)^2) \approx \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} exp(X_i + Y_j)^2$$

em que $X \sim U(0,1)$ e $Y \sim U(0,1)$.

Código R:

```
n <- 10^4
x <- runif(n)
y <- runif(n)
mean(sapply(x,FUN = function(x) exp((x + y)^2)))
[1] 5.017221
```

d) $\int_0^2 exp(x^2/2)dx$

Solução: No item d) devemos fazer a transformação $u = \frac{x}{2}$. E assim, chegaremos a seguinte integral,

$$\int_0^2 exp(x^2/2)dx = 2\int_0^1 exp(2u^2)du = 2E(exp(2U^2)) \approx \frac{2}{n} \sum_{i=1}^{n} exp(2U_i^2).$$

Para encontrar o valor aproximado devemos executar o seguinte código no R,

Código R:

```
n <- 10^7
U <- runif(n)
2*mean(exp(2*U^2))
[1] 4.727395
```

Questão 4.

Solução:

$$\begin{aligned} Cov(U, e^U) &= E(Ue^U) - E(U)E(e^U) \\ &= 1 - \frac{e-1}{2} = \frac{3-e}{2} = 0.14085. \end{aligned}$$

Código R:

```
n <- 10^7
U <- runif(n)
cov(U,exp(U))
[1] 0.1408358
```

Como podemos ver, a aproximação é boa até a quarta casa decimal.

Questão 5.

a) **Solução:**

X	6	4	8	2	10
p	0.40	0.30	0.15	0.10	0.05
$\sum p$	0.40	0.70	0.85	0.95	1

Algoritmo:

* Simule $u \sim U(0,1)$
* Se $u < 40\%$, faça $x = 6$ e pare
* Se $u < 70\%$, faça $x = 4$ e pare
* Se $u < 85\%$, faça $x = 8$ e pare
* Se $u < 95\%$, faça $x = 2$ e pare
* $x = 10$ caso contrário

b) **Solução:** Para n = 50,

	2	4	6	8	10
n = 50	0.12	0.30	0.36	0.14	0.08
valor verdadeiro	0.10	0.30	0.40	0.15	0.05

Note que, para n = 50, as frequências relativas são razoavelmente próximas das probabilidades reais. Podemos verificar isso calculando o desvio absoluto médio das probabilidades, que dá 0.02.

c) **Solução:** Para n = 1000,

	2	4	6	8	10
n = 1000	0.09	0.32	0.40	0.14	0.05
valor verdadeiro	0.10	0.30	0.40	0.15	0.05

Note que, para n = 1000, as frequências relativas são mais acuradas do que com $n = 50$, pois o desvio absoluto médio das probabilidades é de 0.01.

```
set.seed(2020)
RX  <- function(n){ px <- c(0.4,0.3,0.15,0.1,0.05)
                    X <- c(6,4,8,2,10)
                    x <- numeric(n)
                   Fx <- cumsum(px)
                    for(i in 1:n){   U <- runif(1)
                    k = 1
                    while(U > Fx[k]){ k = k + 1     }
                   x[i] = X[k] }
                   tab <- table(x)/n
                   return(tab)}
mean(abs(RX(50) - c(0.1,0.3,0.4,0.15,0.05)));    mean(abs(RX(1000) - c(0.1,0.3,0.4,0.15,0.05)))
```

Questão 6.

Solução: Considere as variáveis $X_1, ..., X_n$, tais que:

$$X = \begin{cases} 1, \text{se ocorreu coincidência na i-ésima carta} \\ 0, \text{c.c} \end{cases}$$

Portanto, temos que $P(X_i = 1) = \frac{1}{n}$ e $P(X_i = 1, X_j = 1) = \frac{(n-2)!}{n!}$. Dessa forma, $X_1, ..., X_n$ são identicamente distribuídos com distribuição $Ber(1/n)$ e **não são independentes**. Assim, $N = \sum_{i=1}^{n} X_i$ é o número de coincidências obtidas e sua média é igual a

$$E(N) = E\left(\sum_{i=1}^{n} X_i\right) = \sum_{i=1}^{n} E(X_i) = \sum_{i=1}^{n} \frac{1}{n} = 1.$$

Como $X_1, \cdots, X_n$ não são independentes, temos que:

$$\begin{aligned} Cov(X_i, X_j) &= E(X_i X_j) - E(X_i)E(X_j) = P(X_i = 1, X_j = 1) - P(X_i = 1)P(X_j = 1) \\ &= \frac{(n-2)!}{n!} - \frac{1}{n^2} = \frac{1}{n^2(n-1)}. \end{aligned}$$

$$\begin{aligned} Var(N) &= Var\left(\sum_{i=1}^{n} X_i\right) = \sum_{i=1}^{n} Var(X_i) + \sum_{i=1}^{n} \sum_{\substack{j=1 \\ i \neq j}} Cov(X_i, X_j) \\ &= \sum_{i=1}^{n} \frac{1}{n}\left(1 - \frac{1}{n}\right) + n(n-1)\frac{1}{n^2(n-1)} = 1 - \frac{1}{n} + \frac{1}{n} = 1. \end{aligned}$$

O problema de embaralhamento das cartas pode ser relacionado ao problema de permutação da sequência $1, 2, \cdots, 100$, uma vez que cada carta é retirada sem reposição. Utilizando o algoritmo descrito em aula para a simulação de permutações aleatórias, temos o seguinte algoritmo para simular o processo de embaralhamento das cartas.

1 Consideramos a sequência $P_1, \cdots, P_{100}$, tal que $P_i = i$, para $i = 1, \cdots, 100$

2 Faça $k = 100$

3 Enquanto $k > 1$ faça

4 Simule $U \sim Unif(0, 1)$ e faça $I = Int[k * u] + 1$

5 Troque os P_I e P_k entre si

6 Faça $k = k - 1$

Código R

```
# Algoritmo

embaralhar <- function(n){
      x <- seq(1:n)
      k <- n
      while(k > 1){
               u <- runif(1,0,1)
               posicao <- floor(k*u)+1
               aux <- x[posicao]
               x[posicao] <- x[k]
               x[k] <- aux
               k <- k-1 }
      return(x)
      }
      a <- seq(1:100)
        c<- vector()
        for(i in 1:1000){
        x <- embaralhar(100)
        c[i] <- sum(x==a)
        }
        > mean(c)
        [1] 0.968
        > var(c)
        [1] 0.989966

#outra maneira
baralho <- 1:100
coincid <- replicate(10^7, sum(sample(baralho,100) == baralho) )

> mean(coincid)
[1] 1.000253
> var(coincid)
[1] 1.000643
```

Como podemos ver, os valores simulados foram próximos dos valores reais.

Questão 9.

a) **Solução:** Pelo método de rejeição,

$$\frac{f(x)}{g(x)} = \frac{\frac{1}{2}x^2e^{-x}}{\lambda e^{-\lambda x}} = \frac{1}{2\lambda}x^2e^{-(1-\lambda)x} \leq C.$$

Fazendo $\frac{d}{dx}\left(\frac{f(x)}{g(x)}\right) = 0$ e simplificando, chegaremos a $2x - (1-\lambda)x^2 = 0$, cujo as raízes são $x_1 = 0$ e $x_2 = \frac{2}{1-\lambda}$. O valor de λ que minimiza o número esperado de iterações do algoritmo é dado minimizando a seguinte função,

$$\frac{f(\frac{2}{1-\lambda})}{g(\frac{2}{1-\lambda})} = \frac{2}{\lambda(1-\lambda)^2}e^{-2} = C(\lambda).$$

Minimizando a função $C(\lambda)$ encontraremos que o mínimo ocorre quando $\lambda = \frac{1}{3}$. Assim, o número esperado de iterações do algoritmo é de $C(\lambda = \frac{1}{3}) = 1.83$ e a probabilidade média de rejeição é $1 - \frac{1}{C(\lambda=\frac{1}{3})} = 1 - \frac{1}{1.83} = 0.453$.

b) **Solução:** Temos que a média das distribuições $Gama(3,1)$ e $Exp(1/3)$ são iguais a 3, ou seja, a exponencial que minimiza o número médio de iterações necessárias para gerar uma variável aleatória $Gama$ tem a mesma média da distribuição $Gama$ a ser simulada.

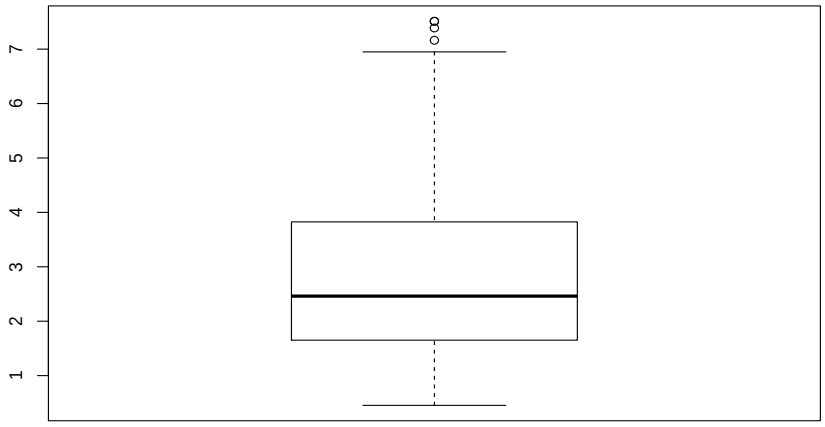
c) **Solução:** Com $C = C(1/3) = 1.83$, $f(y) = \frac{1}{2}x^2e^{-x}$ e $g(y) = \frac{1}{3}e^{-\frac{x}{3}}$, temos

Algoritmo:

Passo 1: Simule $Y \sim Exp(1/3)$

Passo 2: Simule $U \sim U(0,1)$

Passo 3: Se $U \leq \frac{f(y)}{Cg(y)}$, faça $X = Y$. Caso contrário, retorne ao passo 1



Código R

```
f <- function(y){ 1/2*y^2*exp(-y)}
g <- function(y){ 1/3*exp(-1/3*y)}
x <- c()
C <- 2/(1/3*(1-1/3)^2*exp(2))
n <- 1
while (n <= 100) {   y <- rexp(1,1/3)
                     u <- runif(1)
                     if(u <= f(y)/(C*g(y))){ x[n] = y
                                             n = n+1
                                          }
}
boxplot(x)
```