<u>Question 1:</u>

See files in directory Task 1/
Please compile withdrawal.c like so: "gcc withdrawal.c -lm ranlxd.c"
Note: this simulation may take a minute to complete.

To answer this question I simulated the total amount withdrawn per month over many months.

This required exponentially and poisson distributed RVs. Methods described in *Simulation 3rd edition* by Sheldon Ross, pages 51 and 65 , were used to create RVs with these distributions using the RVs generated by RANLUX.

I simulated the amount withdrawn per month for 1'000'000 months and found that 10.97% of the months had over 50'000 euro withdrawn in total. Therefore the probability that the total amount withdrawn per month is over 50'000 is 0.1097.

In addition I recorded the simulated average amount per withdrawal, and the simulated average number of withdrawals per month. The simulated average number of withdrawals per month was 50.002779, and the simulated average amount per withdrawal was 800.114457. These figures are very close to the given figures, which makes me believe my simulation was accurate.

<u>Question 2:</u>

See files in directory Task 2/
Please compile integral_estimation.c and integral_estimation_exponential.c like so:
"gcc <source>.c -lm ranlxd.c"

(a):

Because the integral is from 0 to 3 to be able to use RVs generated by RANLUX we need to adjust the integral so that it is from 0 to 1. This was done using the substitution method described in *Simulation 3rd edition* by Sheldon Ross, page 39.

The results were as follows:

N	Result	Absolute error
100	0.858712	0.014397
1'000	0.859169	0.013940
10'000	0.875482	0.002373
100'000	0.873640	0.000531
1'000'000	0.873082	0.000027

(b):

The inverse transform method can turn the RVs generated by RANLUX (uniform distribution across [0,1)) into exponentially distributed RVs. As per *Simulation 3rd edition* by Sheldon Ross page 64 Example 5b, by generating a RV *U* with RANLUX we can generate an exponentially distributed RV *X* by setting *X* = $(-\frac{1}{\lambda})log(U)$, where λ is the rate of *X*.

I performed experiments to find a suitable value for λ. I found that a value of ~0.452 usually gave the best results, though this depends on the exact RVs generated in each run. Setting λ above 1 gave very poor results.

Results with λ = 0.452:

N	**Result**	**Absolute error**
100	0.851819	0.021290
1’000	0.867744	0.005365
10’000	0.868344	0.004765
100’000	0.869986	0.003123
1’000’000	0.873518	0.000409

Question 3:

(a):

First with p = 0.4, n = 20, a = 16:

Original mean and variance:
The original mean is n*p, which is 20*0.4 = 8
The original variance is n*p*(1-p), which is 20*0.4*(1-0.4) = 4.8

$$E_{ft}[S] = E_{ft}[\sum_i X_i] = \sum_{i=1}^{n} \frac{p_i e^t}{p_i e^t + (1-p_i)} = n(\frac{pe^t}{pe^t + (1-p)})$$

$$E_{ft*}[S] = 20(\frac{0.4e^{t*}}{0.4e^{t*} + 0.6}) = 16 = a$$

$$\Rightarrow e^{t*} = 6 \; and \; p' = 0.8$$

$$M(t*) = (e^{t*}p + (1-p))^n = (0.4e^{t*} + 0.6)^{20} = 3^{20}$$

$$e^{-t*S} = (\frac{1}{6})^{16}$$

$$\widehat{\theta} = I\{S \geq 16\}(\tfrac{1}{6})^{16}3^{20}$$
$$\widehat{\theta} = (\tfrac{1}{6})^{16}3^{20}$$

$$\theta = E_f[I\{S \geq a\}] = \sum_{k=16}^{20} pk \approx 0.000317$$
$$where\, pk = \binom{n}{k}p^k(1-p)^{n-k}$$
$$Var_f[I\{S \geq 16\}] = \theta(1-\theta) \approx 3.1639 \times 10^{-4}$$
$$\theta \leq \widehat{\theta} \leq M(t*)e^{-ta} = (\tfrac{1}{6})^{16}3^{20} \approx 0.001236 = c$$

$$E_{ft}[\widehat{\theta}] = \sum_j \widehat{\theta}_j\, p_j \leq c\sum_j p_j = c$$

$$E_{ft}[\widehat{\theta}^2] = \sum_j \widehat{\theta}_j^{\,2}\, p_j \leq c\sum_j \widehat{\theta}_j p_j = cE_{ft}[\widehat{\theta}]$$

$$Var_{ft}[\widehat{\theta}] = E_{ft}[\widehat{\theta}^2] - E_{ft}[\widehat{\theta}]^2 \leq cE_{ft}[\widehat{\theta}] - E_{ft}[\widehat{\theta}]^2$$
$$= c^2(\tfrac{E_{ft}[\widehat{\theta}]}{c})(1 - \tfrac{E_{ft}[\widehat{\theta}]}{c}) \leq \tfrac{c^2}{4} = 3.81924 \times 10^{-7}$$

<u>Now with p = 0.2, rest unchanged:</u>

Original mean and variance:
The original mean is n*p, which is 20*0.2 = 4
The original variance is n*p*(1-p), which is 20*0.2*(1-0.2) = 3.2

$$E_{ft}[S] = E_{ft}[\sum_i X_i] = \sum_{i=1}^{n} \frac{p_i e^t}{p_i e^t + (1-p_i)} = n(\frac{pe^t}{pe^t + (1-p)})$$
$$E_{ft*}[S] = 20(\frac{0.2e^{t*}}{0.2e^{t*} + 0.8}) = 16 = a$$
$$\Rightarrow e^{t*} = 16\, and\, p' = 0.8$$
$$M(t*) = (e^{t*}p + (1-p))^n = (0.2e^{t*} + 0.8)^{20} = 4^{20}$$
$$e^{-t*S} = (\tfrac{1}{16})^{16}$$
$$\widehat{\theta} = I\{S \geq 16\}(\tfrac{1}{16})^{16}4^{20}$$
$$\widehat{\theta} = (\tfrac{1}{16})^{16}4^{20}$$

$$\theta = E_f[I\{S \geq a\}] = \sum_{k=16}^{20} pk \approx 1.38035 \times 10^{-8}$$

$$Var_f[I\{S \geq 16\}] = \theta(1-\theta) \approx 1.38034998 \times 10^{-8}$$

$$\theta \leq \widehat{\theta} \leq M(t*)e^{-ta} = (\tfrac{1}{16})^{16}4^{20} \approx 5.9 \times 10^{-8} = c$$

$$E_{ft}[\widehat{\theta}] = \sum_j \widehat{\theta}_j p_j \leq c\sum_j p_j = c$$

$$E_{ft}[\widehat{\theta}^2] = \sum_j \widehat{\theta}_j^2 p_j \leq c\sum_j \widehat{\theta}_j p_j = cE_{ft}[\widehat{\theta}]$$

$$Var_{ft}[\widehat{\theta}] = E_{ft}[\widehat{\theta}^2] - E_{ft}[\widehat{\theta}]^2 \leq cE_{ft}[\widehat{\theta}] - E_{ft}[\widehat{\theta}]^2$$

$$= c^2(\tfrac{E_{ft}[\widehat{\theta}]}{c})(1 - \tfrac{E_{ft}[\widehat{\theta}]}{c}) \leq \tfrac{c^2}{4} = 8.702510^{-16}$$

(b):

See file Task 3/bernoulli_importance.c.
Please compile like so: gcc bernoulli_importance.c ranlxd.c -lm

My simulation makes use of p' to sample the distribution so that the expected value is 16. For both p=0.4 and p=0.2 the estimator variance was slightly below the maximum I calculated above, but still within the same order of magnitude. Doing 10'000 simulations with a = 16, n = 20 I got the following results:

	With p=0.4	**With p=0.2**
e^{t*}	6	16
p'	0.8	0.8
M(t*)	3^{20}	4^{20}
Average sum	15.9757	16.0036
Average estimator	0.00030332	1.37239×10^{-8}
Estimator std dev	0.0004642	2.574343×10^{-8}
Estimator variance	2.155615×10^{-7}	$6.62724286 \times 10^{16}$

Question 4:

(a):

χ_1 = "pizza" =

1
0

χ_2 = "burrito" =

0
1

M_{22} = P(pizza tomorrow | pizza today) = 0.15
M_{11} = P(burrito tomorrow | burrito today) = 0.3
M_{12} = P(burrito tomorrow | pizza today) = 0.85
M_{21} = P(pizza tomorrow | burrito today) = 0.7

M =

0.3	0.85
0.7	0.15

(b):

Pizza on Sunday => P_0 =

1
0

Monday => $P_1 = M*P_0$ =

0.3
0.7

Tuesday => $P_2 = M*P_1$ =

0.685
0.315

Probability of pizza on Tuesday given pizza on Sunday is 68.5%

(c):

Finding the eigenvalues and eigenvectors of M gives:

$\lambda_1 = 1, \quad \lambda_2 = \frac{-11}{20}$

v_1 =

17/14
1

v_2 =

-1
1

Using these eigenvalues and eigenvectors we get:

U =

17/14	-1
1	1

U^{-1} =

14/31	14/31
-14/31	17/31

D =

1	0
0	-11/20

By writing M as UDU^{-1} we can expand $P_N = MP_{N-1}$ to[1]:

$P_N = UD^NU^{-1}P_0$

by cancelling out every occurrence of $U^{-1}U$ in the middle part.

As D is a diagonal matrix we can write D^N as:

$(1)^N$	0
0	$(-11/20)^N$

So as n approaches infinity D^N becomes:

1	0
0	0

[1] MIT OpenCourseWare: https://www.youtube.com/watch?v=nnssRe5DewE at time 6:14

Solving $P_N = UD^NU^{-1}P_0$ as n approaches infinity gives:
P_N =

17/31
14/31

The columns here add to 1 as would be expected. This shows that over a long term for dinner the student will eat a pizza 54.84% of the time, and a burrito 45.16% of the time.