Q1)

a) since t_A and t_B is uniformly distributed on $[0, 100]$, ($a = 0$, $b = 100$)

$f(t_A) = 1/(b-a) = 1/100$

$f(t_B) = 1/(b-a) = 1/100$

$f_{t_A, t_B}(t_A, t_B) = f(t_A) * f(t_B) = \underline{1/10^4}$

and the CDF is $F_{t_A, t_B}(t_a, t_b) = \iint f_{t_A, t_B}(t_a, t_b)\, dt_A\, dt_B$

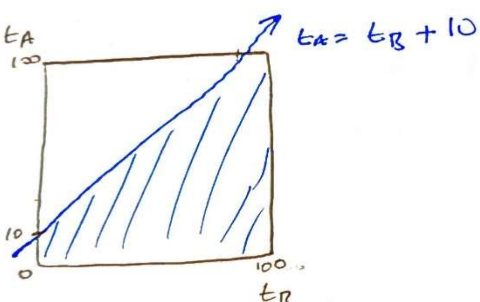
$= \underline{\dfrac{t_A \cdot t_B}{10^4}}$

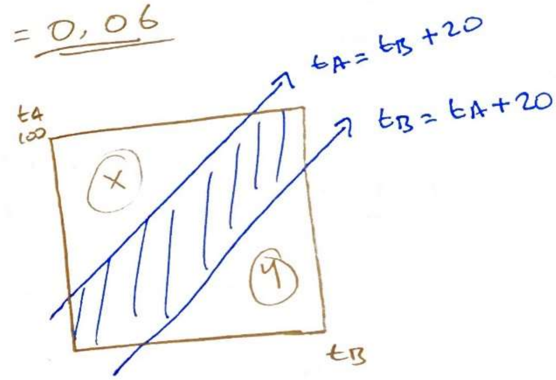
b) $P\{T_A \leq 30 \cap 40 \leq T_B \leq 60\}$

$P\{T_A \leq 30\} \times (P\{T_A \leq 60\} - P\{T_A \leq 40\})$

$= 3/10 \times (6/10 - 4/10) = \underline{0,06}$

c)





The probability is equal to the marked area / total area

$1 - \dfrac{90 \times (90/2)}{10^4}$

$= \underline{0.595}$

$= 1 - (x + y)$

$= 1 - \dfrac{80 \times 80}{10^4}$

$= \underline{0,36}$

Q2) $n = 150$

a) $p = 0,6$

$q = 0,4$

$\sigma = \sqrt{n * p * q}$

$= \sqrt{36}$

$= \underline{6}$

$\mu = n * p$

$= 90$

We are asked to find $P(x \geq 98)$ (at least %65 of the shoppers $= \underline{97.5}$)

$$z = \frac{x - \mu}{\sigma} = \frac{98 - 90}{6} = 1,33$$

$$P(x \geq 98) = P(z \geq 1,33) = \underline{0,0918}$$

At most %15 of customers are freq shoppers, $\underline{P(x \leq 22)}$

b) $\mu = 150 * 0,1 = 15$

$\sigma = \sqrt{13,5}$

$= 3,674$

$$z = \frac{x - \mu}{\sigma} = \frac{22 - 15}{3,674} \cong 1,9$$

$$P(x \leq 22) = P(z \leq 1,9) = \underline{0,9713}$$

Q3) First, we have to convert the height values into z-scores.

$Z = (X - \mu) / \sigma$

for $X_1 = 170$ cm

$Z_1 = (170 - 175)/7 \approx -0,714$

for $X_2 = 180$ cm

$Z_2 = (180 - 175)/7 \approx 0,714$

$P(Z \leq -0,714) \approx 0,2389$

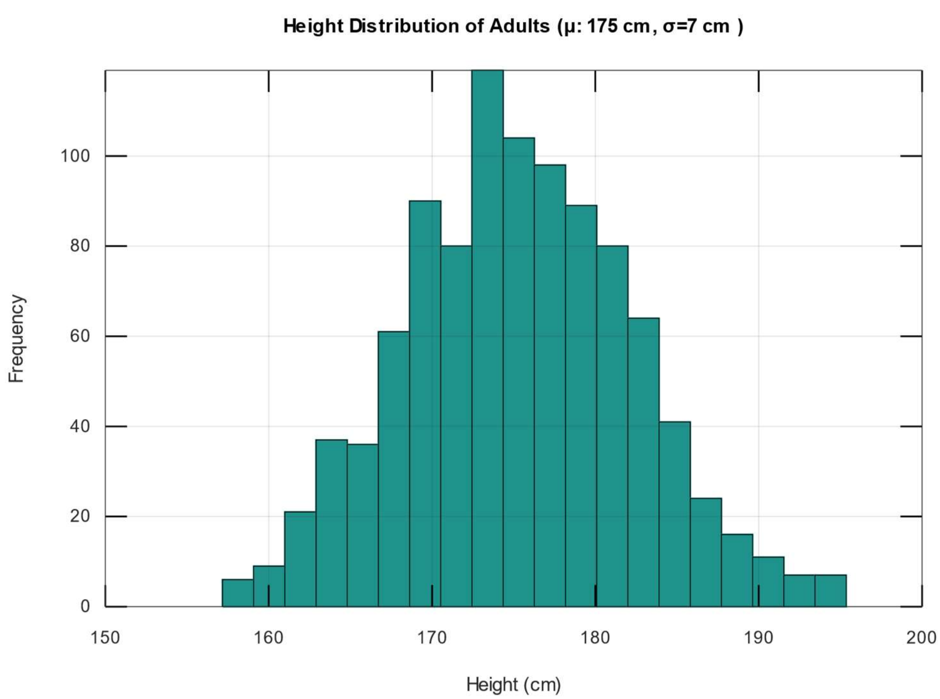
$P(Z \leq 0,714) \approx 0,7611$

So, $P(170 \leq \text{height} \leq 180)$

$= P(Z \leq 0,714) - P(Z \leq -0,714)$

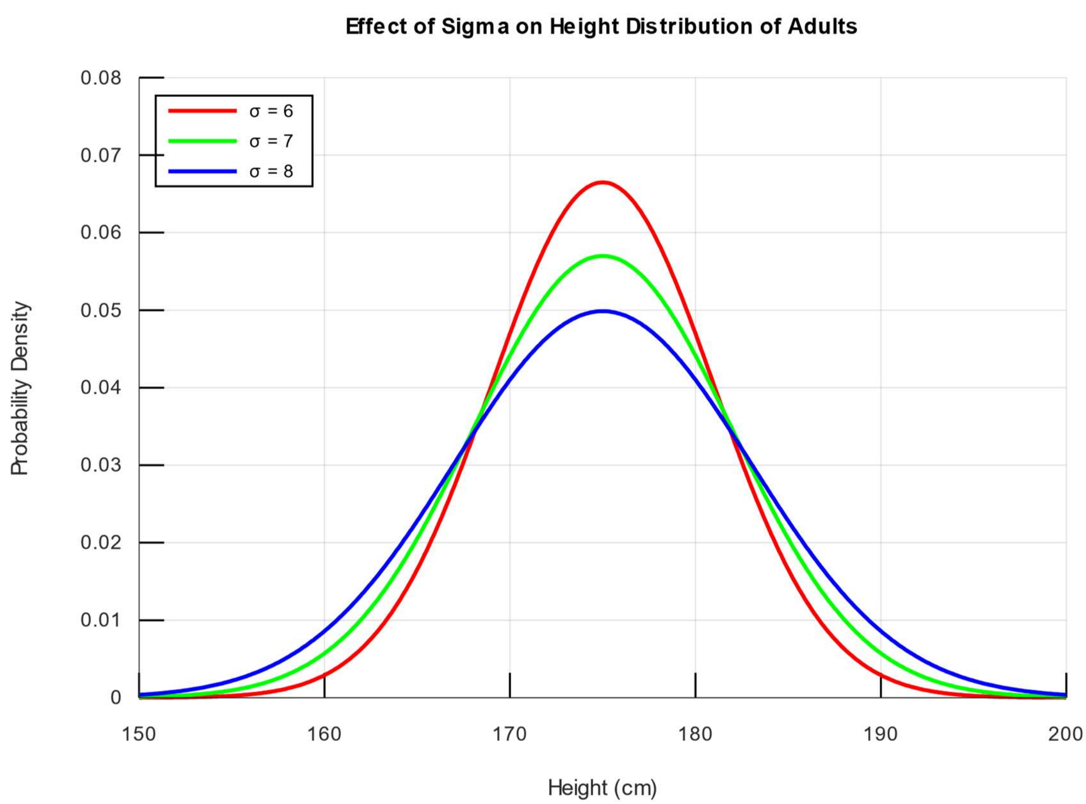
$= 0,7611 - 0,2389 = \underline{0,522}$

Q4-a)



```
mu = 175;

sigma = 7;

num_samples = 1000;

heights = normrnd(mu, sigma, 1, num_samples);

hist(heights, 20); % You can adjust the number of bins with the second parameter

title("Height Distribution of Adults (µ: 175 cm, σ=7 cm )");

xlabel("Height (cm)");

ylabel("Frequency");

grid on;

axis([150, 200, 0, inf]); % Set the axis limits for better visualization
```

Q4-b)




```
%Q4b visualize the effect of sigma on the distribution by plotting the probability
density function (PDF) of the normal distribution for different values of sigma: 6, 7,
and 8.

mu = 175;

sigmas = [6, 7, 8];
```

```
height_range = 150:0.1:200;

pdf_values = zeros(numel(sigmas), numel(height_range));

for i = 1:numel(sigmas)

  pdf_values(i, :) = normpdf(height_range, mu, sigmas(i));

endfor

figure;

hold on;

colors = ["r", "g", "b"];

for i = 1:numel(sigmas)

  plot(height_range, pdf_values(i, :), 'Color', colors(i), 'LineWidth', 2,
'DisplayName', ['\sigma = ' num2str(sigmas(i))]);

endfor

title("Effect of Sigma on Height Distribution of Adults");

xlabel("Height (cm)");

ylabel("Probability Density");

% Add a legend and grid to the plot

legend('show', 'location', 'northwest');

grid on;

hold off;
```

Q4-c)

```
%Q4c-simulate and estimate the probability of having at least 45%, 50%, and 55% of
adults with heights between 170 cm and 180 cm.

function heights = generate_heights(mu, sigma, num_samples)

  heights = normrnd(mu, sigma, 1, num_samples);

endfunction

mu = 175;

sigma = 7;

lower_height = 170;

upper_height = 180;

num_samples = 150;

num_iterations = 1000;

thresholds = [0.45, 0.5, 0.55];

count_in_range = zeros(1, numel(thresholds));


for i = 1:num_iterations
```

```
  heights = generate_heights(mu, sigma, num_samples);


  in_range = sum(heights >= lower_height & heights <= upper_height) / num_samples;


  % Update the counter array for each threshold
  for j = 1:numel(thresholds)
    if in_range >= thresholds(j)
      count_in_range(j) += 1;
    endif
  endfor
endfor
% Calculate the probability array based on the simulation results
prob_simulation = count_in_range / num_iterations;
for i = 1:numel(thresholds)
  printf("The probability of having at least %.0f%% of adults with heights between %d cm and %d cm, estimated from %d iterations, is %.4f.\n", thresholds(i) * 100, lower_height, upper_height, num_iterations, prob_simulation(i));
endfor
```