Step 1

Claim: $count(n,B) \le \overline{count(n,\ n^2)}$

Proof:

By definition of a feasible knapsack solution we have: $\sum_{i=1}^{n} w_i y_i \le B$.

Additionally, we define $a_i = floor(\frac{n^2 w_i}{B})$.

Therefore, $\overline{count(n,n^2)}$ belongs to the set $\left\{ y \in \{0,1\}^n \ : \ \sum_{i=1}^{n} a_i y_i \le n^2 \right\}$.

We can prove that $count(n,B)$ is a subset of $\overline{count(n,n^2)}$ by showing that

$$
\begin{aligned}
\sum_{i=1}^{n} w_i y_i \le \sum_{i=1}^{n} a_i y_i & \\
= \sum_{i=1}^{n} floor(\tfrac{n^2 w_i}{B}) y_i & \qquad floor(\tfrac{n^2 w_i}{B}) \le \tfrac{n^2 w_i}{B} \\
\le \tfrac{n^2}{B} \sum_{i=1}^{n} w_i y_i & \qquad \sum_{i=1}^{n} w_i y_i \le B \\
\le n^2 &
\end{aligned}
$$

Therefore, we have $\sum_{i=1}^{n} w_i y_i \le B \le n^2$. The number of solutions to $\overline{count(n,\ n^2)}$ is at least as big as the number of solutions to $count(n,B)$.

Step 2

Let f be a function from the set of rounded solutions *S'* to the set of actual solutions *S*. Then $f(S) = S$ by definition. We need to show that f(S') maps to at most (n+1) solutions to f(S).

The rounded solution is defined as $\sum_{i=0}^{n} a_i y_i \le n^2$. We have a definition of $a_i = floor\left(\frac{n^2 w_i}{B}\right)$. Let us define the floor error as $\triangle_i = a_i - w_i \frac{n^2}{B}$. And we know that the the floor function will cause a difference of at most 1 for each $\triangle_i$.

Let us take $C = S' - S$, the set of the extra solutions to the rounded version of the problem. Then we need to show that f(C) maps to at most (n+1) solutions to S.

It follows that each solution to C must have an error of at most 1 from the actual solution and there are at most n such solutions since there are n a_i elements. Therefore, the number of solutions to the rounded version of the problem will map to at most (n+1) solutions to S.

Step 3

Pseudo-code from page 4 of the coursework instruction sheet runs in $\Theta(n^3)$.

Analysis:

Line 1 takes $\Theta(1)$ with the correct data structure that keeps track of the size. Lines 2-3 are executed n times and do $\Theta(1)$ work at each step → $\Theta(n)$. Line 4 takes $\Theta(n^3)$ to build and dominates the runtime. Line 5 is executed in constant time. Therefore, runtime of the Approximation algorithm is $\Theta(n^3)$

Reason for $\Theta(n^3)$ runtime of *CountKnapsackDP(a,n²)*:

We have $length(w) = n$, $B = n^2$.

Line 1 of CountKnapsackDP takes Θ(1), line 2 takes Θ(n+1)Θ(n^2+1) to allocate. Line 4-9 are executed n times and lines 5-8 are executed exactly n^2 times giving us combined runtime of $\Theta(n^3)$. Line 10 runs in constant time. Therefore, *CountKnapsackDP(a,n²)* will take $\Theta(n^3)$ to run.