1. **(2 points) From the moment that g receives her first proposal, g remains engaged. Also, her sequence of partners gets better and better (in terms of her list of preferences).**

   From problem 1.20 solution in the text.

   b_{s+1} proposes to the girls according to his list of preference; a g ends up accepting, and if the g who accepted b_{s+1} was free, she is the new one with a partner. Otherwise, some $b^* \in \{b_1, ..., b_s\}$ became disengaged, and we repeat the same argument. The g's disengage only if a better b proposes, so it is true that $p_{M_{s+1}}(g_j) <^j p_{M_s}(g_j)$.

2. **(2 points) The sequence of g's to whom a particular b proposes gets worse and worse (again in terms of his list of preferences).**

   From page 15 and problem 1.19 solution in the text.

   When b_{s+1} proposes to a girl g_j, g_j accepts his proposal if either g_j is not currently engages or is currently engaged to a less preferable boy b, i.e. $b_{s+1} <^j b$. In the case where g_j prefers b_{s+1} over her current partner b, then g_j breaks off the engagement with b and b then has to search for a new partner.

   After b proposed to g for the first time, whether this proposal was successful or not, the partners of g could have only gotten better. Thus, there is no need for b to try again.

   It follows that the sequence of g's to whom a particular b proposes gets worse and worse.

3. **(2 points) The following is an invariant of the G-S Algorithm: *if b is free (not engaged) at some point in the execution of the algorithm, then there is a g to whom he has not yet proposed.***

   Proof by contradiction.

   Assume to the contrary that there exists a boy b_i that is not engaged and has proposed to all g on his list at some stage in the execution. This means that b_i was rejected by every g on his list, which implies that each g on his list was already engaged to a more preferable b. This contradicts our original assumption because we have n girls engaged to $n - 1$ boys and there are no free girls.

4. **(2 points) The set of pairs M at the end of the execution of the algorithm constitutes a *perfect matching.* (Start by defining what a "perfect matching" is.)**

   Proof by contradiction.

   A perfect matching is one in which every boy b is uniquely engaged to a girl g, there exists a $1:1$ matching between the boys and girls. Assume that this is not the case at the end of the execution of the algorithm. Considering the algorithm will not allow a g to engage to a b without first being free (disengaging if necessary), there will never be a case with a one to many relationship. This implies that there exists a girl g_j and a boy b_i that are free at the end of the execution of the algorithm. If g_j is free, that means that she was never proposed to, but since b_i would have proposed to all g on his list, then he must have proposed to g_j. This yields a contradiction and it follows that a perfect matching must exist.

5. **(2 points) The set of pairs M at the end of the execution of the algorithm constitutes a *stable matching.* (Start by defining what a "stable matching" is.)**

   From problem 1.21 solution in the text.

   A matching M is stable if it contains no blocking pairs.

   Suppose that we have a blocking pair $\{b, g\}$ (meaning that $\{(b, g'), (b', g)\} \subseteq M_n$, but b prefers g to g', and g prefers b to b'). Either b after b' or before. If b came before b', then g would have been with b or someone better when b' came around, so g would not have become engaged to b'. On the other hand, since (b', g) is a pair, no better offer has been made to g after the offer of b', so b could not have come after b'. In either case we get an impossibility, and so there is no blocking pair $\{b, g\}$, and thus, the matching is stable.

6. **(2 points) Give an example of a B, G with corresponding lists of preferences for which there is more than one stable matching.**

   Boy optimal and girl optimal run on the same input.

   Take $B = \{b_1, b_2\}$ such that $g_2 <_1 g_1$ and $g_1 <_2 g_2$ and $G = \{g_1, g_2\}$ such that $b_1 <^1 b_2$ and $b_2 <^2 b_1$.

   Boy optimal $M = \{(b_1, g_2), (b_2, g_1)\}$

   Girl optimal $M' = \{(b_1, g_1), (b_2, g_2)\}$

7. **(2 Points) Recall the definition of a *feasible pair* in the textbook (pg. 17). Let's say that g is the *best feasible pair* for b, if (b, g) is a feasible pair, and there is no g' such that:**

   $g' <_b g$ **and** (b, g') **is also a feasible pair.**

   For any given b, let $\mathcal{B}(b)$ be b's best feasible pair. Finally, let $M^* = \{(b, \mathcal{B}(b)) : b \in B\}$. Show that the G-S Algorithm yields M^*.

   From problem 1.22 solution in the text, proving the algorithm is boy-optimal.

   Assume to the contrary that b is the first boy who is rejected by his best feasible partner $\mathcal{B}(b)$. This means that $\mathcal{B}(b)$ has already been paired with some b', and $\mathcal{B}(b)$ prefers b' to b. Furthermore, $\mathcal{B}(b)$ is at least as desirable to b' as his own best feasible partner (since the proposal of b is the first time during the run of the algorithm that a boy is rejected by his best feasible partner). Since $\mathcal{B}(b)$ is the best feasible partner for b we know (by definition) that there exists some stable matching S where $(b, \mathcal{B}(b))$ is a pair. On the other hand, the best feasible partner of b' is ranked (by b' of course) at most as high as $\mathcal{B}(b)$, and since $\mathcal{B}(b)$ is taken by b, whoever b' is paired with in S, say g', b' prefers $\mathcal{B}(b)$ to g'. This gives us an blocking pair because $\{b', \mathcal{B}(b)\}$ prefer each other to the partners they have in S. Therefore, it follows that each $b \in B$ must be paired with $\mathcal{B}(b)$, their best feasible partner.

8. **(2 points) Show that any re-ordering of B still yields M^*, that is, the G-S Algorithm is independent of the order of the boys.**

   From problem 1.22 solution in the text.

   From problem 7 it can be seen that the ordering of the boys is immaterial, because there is a unique boy-optimal matching (each boy b is paired with $\mathcal{B}(b)$), and it is independent of the ordering of the boys.

9. **(2 points) Show that in M^*, each g is paired with her worst feasible partner.**

   From problem 1.22 solution in the text, proving the algorithm is girl-pessimal.

   To show that the Gale-Shapley algorithm pairs each girl with her worst feasible partner, we use the fact that each boy is paired with his best feasible partner (which we just showed in problem 7). Again, we argue by contradiction. Suppose there is a stable matching S where g is paired with b, and g prefers b' to b, where (b', g) is the result of the Gale-Shapely algorithm. By the fact that each boy is paired with their best feasible partner, we know that in S we have (b', g'), where g' is not higher on the preference list of b' than g, and since g is already paired with b, we know that g' is actually lower. This says that S is unstable because $\{b', g\}$ form a blocking pair as they would rather be together than with their partners.

10. **(2 points) Access the running time (complexity) of the algorithm in terms of Big-Oh complexity.**

    See page 16 in the text.

    The algorithm runs in $O(n^3)$ time. This is because at any stage $s+1$ there is an upper bound of $(s+1)^2$ steps (this is the case where a g is previously engaged and forces another b to search for a new partner), giving a time complexity of $O(n^2)$ per stage. And, since there are n stages total, this yields a final time complexity of $O(n^3)$.