Reading:

The goal is to finish Chapter 11 by the end of spring break and then continue into quantum computation.

(1) For any a, $0 < a < 1$, define PP_a to be the class of languages L such that for some polynomial $p(n)$ and predicate $R(x,y)$ decidable in time $p(|x|)$, and all x,

$$x \in L \iff \Pr_{|y|=p(|x|)}[R(x,y)] > a.$$

Show that $\mathrm{PP}_a = \mathrm{PP}$. Does this hold if $a = a(n)$ is given by an inverse polynomial function $a(n) = 1/q(n)$? How about if $q(n) = 2^n$? How slow-growing can $a(n)$ be to make this work?

(2) Now given $0 < a < b < 1$, define $\mathrm{BPP}_{a,b}$ to be the class of languages L such that for some $p(n)$ and $R(x,y)$ as above, and all x:

$$x \in L \implies \Pr_y[R(x,y)] \geq b;$$
$$x \notin L \implies \Pr_y[R(x,y)] \leq a.$$

(Here I've left tacit that y ranges over $\{0,1\}^{p(|x|)}$.) Show that $\mathrm{BPP}_{a,b} = \mathrm{BPP}$. But now for the real question: Suppose a and b depend on n as in the final part of problem (1). Most in particular, suppose $q(n)$ and $q'(n)$ are polynomials such that $a(n) = 1/q(n)$ and $b(n) = a(n) + 1/q'(n)$. Then when you do $t(n)$-many trials to amplify the success probability, do you get a higher power of $q(n)$ versus $q'(n)$, or are they about the same?

(3) Define $\mathcal{U}$ to eb the class of languages L such that for some $p(n)$ and $R(x,y)$ as above, and all x,

$$x \in L \iff (\exists! y)R(x,y).$$

The concept to come in section 11.1 is more stringent in requiring L to "promise" that the case where $R(x,y_1)$ and $R(x,y_2)$ hold with $y_1 \neq y_2$ never happens. Here in that case $x \notin L$.

Does $\mathcal{U}$ contain either NP or co-NP? Can you place $\mathcal{U}$ within the second or third level of the polynomial hierarchy? Is $\mathcal{U}$ closed under complements? After answering these warmup questions, show that if $\mathcal{U} \subseteq \mathrm{BPP}$, then $\mathrm{NP} = \mathrm{RP}$.

(4) [A problem about oracle circuits and relativized SAT. Oracle circuits have k-ary *oracle gates* g for arbitrary k (depending on the input length n) such that if $a = a_1 \cdots a_k$ are the binary inputs to g and $A \subseteq \{0,1\}^*$ is the oracle language, then $g(a)$ returns 1 iff $a \in A$. SAT^A has oracle clauses $(u_1, \ldots, u_k)$ with $u_i = \pm a_i$ for each i that are true iff the assignment makes the signed value string of the clause belongs to A.]