Joshua Hook.

German verb conjugator

Yes Good!

This project will be a tool for users to generate a table containing the inflectional paradigm for any verb in German. It will accept any valid citation form (in German this is the infinitive) of a German verb return the full paradigm of the verb. ✓

German verbs inflect for the following forms, which will all need to be captured by my conjugator:

- Person/number: 1st singular, 1st plural, 2nd singular, 2nd plural informal, 2nd singular/plural formal, 3rd singular, 3rd plural
- Tense/aspect: present, imperfect, perfect, pluperfect, future (I and II)
- Mood: indicative, subjunctive (I and II), imperative ✓

This program will need to take the following aspects of German verb morphology into account:

- Irregular "strong" verbs – these verbs have three irregular stems for certain present tense forms, all imperfect tense forms, and the perfect participle. ✓
- "Separable" verbs – these verbs are comprised of two parts – the base verb and an appended prepositional particle – which are separated in most forms; when the verb is in any position other than final, the prepositional element remains separated at the end of the clause. Certain prefixes appear between this particle and the verb.
- In the perfect tense, a handful of verbs (typically verbs of movement) employ the use of *sein* ("to be") as an auxiliary verb as opposed to *haben* ("to have").
- For some verbs, syllable structure restrictions requires that an epenthetic *-e-* is inserted between the verb stem and certain personal endings.

In terms of the data required, I will need to create a dictionary of German strong verbs containing the different verb stems for each. For this I will need to search for a list of such verbs and their stems. I have ready access to several free online German dictionaries, such as Duden, Leo, and even Wiktionary. I can use these websites to easily obtain information on a given verb's conjugation and stems, and to thereby compare the expected results against my program's actual output. I can likewise create a set of verbs which take *sein* as an auxiliary in the perfect tense. ✓

easy to extract + save in text file?

A tentative list of important functions might be:

- `vb_validate(input)` – validate the user's input, reject if the input is not a valid verbal infinitive ✓
- `get_stem(input, stem)` – looks up a strong verb in the dictionary and returns the specified stem, returns a regularly derived weak verb stem if no strong verb found ✓
- `generate_table()` – generates the full conjugation table as an output ✓
- `conjugate_present(input)` – returns a list of present tense forms for an inputted verb; similar functions would exist for each tense and mood combination, though alternatively this could be handled in a single large function. ✓

a balance to be struck
small functions are usually more re-usable and easier to write

Admittedly, an approach to testing the software using `assert()` is going to require some additional thought. What I am likely to do is create a limited set of test cases covering all special cases and rules I might need to consider. For the final table, I can ensure I get the correct layout by using placeholder values corresponding to each individual cell of the table; if these values display in the correct places, then the table works as intended and can be filled with the actual values. The final tables will be compared against conjugation tables provided by online dictionaries to ensure that I haven't missed anything. ✓

In terms of the software I intend to use for this project, I will use IDLE to create the program, and LaTeX to write/compile the project diary to go with it.

Overall, a ~~decent~~ good! project proposal.
get_stem() might turn out to have a lot of depth ~~that would~~ that needs to be explored – that would be good.

Your definition of project is detailed, I like that.

or maybe there is a better way to test?
I don't know but if assert() is ~~not~~ making ~~it easier~~ testing difficult try another tack.

[illegible]