**How to use this Tibetan sorting code?**

**1. Single word(not useful)**

There is a single php file called WordGranules.php. You need to include this in your php code. This takes in an input of a single Tibetan word, like བཀྲིས་, ཁ or ཙམ. In your php code, to prepare བཀྲིས་ for sorting:

```
include_once 'WordGranules.php';
$eachWord= "བཀྲིས་";
$wordClass=new WordGranules();
$output[$eachWord]= $wordClass→getOrderedParts($eachWord);
```

This will return some Tibetan value that is ready for very simple Unicode sorting.
So, you can use php built-in sort functions like asort rsort like:

```
asort($output);
```

**2. Sorting an array**

Now, you have an array of Tibetan text to sort of course.

```
include_once 'WordGranules.php';
$input_array; ##make sure assign an array, don't declare like this
for ($i=0; $i< count($input_array); $i++) {
   #split text into word by ཨོཾ "", use multi byte since Tibetan is not
single byte romanization
   $wordArray=mb_split("", $input[$i]);
   ##this collects back the prepared words for each of the array element
   $reordVal="";
   foreach ($wordArray as $eachWord) {
      $wordClass=new WordGranules();
      $reordVal.= $wordClass->getOrderedParts($eachWord);
   }
   ##we need indexes attached for sorting multiple occurrence of the
same text
   $output[$i. "-". $input[$i]] =$reordVal;
}
#now asort the new array
asort($output);
##strip off the index attached on each sorted keys
$sortedBo=array();
foreach ($output as $key => $value) {
   $sortedBo[]=mb_split("-", $key)[1];
}
echo "Finally sorted Tibetan: <br>";
print_r($sortedBo);
```

Here is a demo: **http://tennom.tv/php/sort/boSort.php**

**3. insert program**
please follow: **http://sort.000webhostapp.com/insertItem.php**

**How does it work?**
**1. Analyzing parts**
At the initiation, a Tibetan word, which may contain some punctuation, will be split to parts, namely root, suffixes, post-suffixes, vowels, prefixes, sub-scripts, Wazhur, and supper scripts. Each of the characters in a word is analyzed and assigned a part.
**2. Assigning weights**
After that, it will attach a weight at the beginning of each of these detected parts, except for the root letter. Now all of these weight attached parts reordered according to the weights. The higher the weight is the closer a part is ordered after the root letter. For instance, when བསྒྲིགས་ is analyzed, root is ག, the next heaviest part is the supper script ས , then sub-scripts ྒ, prefix བ ,vowel ི, post-suffix ས and finally suffix ག. Consequently, when this word is fed in to the PHP asort function, it's virtually like this, except that the weights are not numerical.
ག 7 ས 6 ྒ 4 བ 3 ི 2 ས 1 ག
The ordering according to weights helps བསྒྲིགས་ to have a higher order than any of བཀྲིགས་(missing 7 ས) བགྲིགས་(missing 7 ས 6 ྒ) གྲིགས་(missing 7 ས 6 ྒ 4 བ) གགས་(missing 7 ས 6 ྒ 4 བ 3 ི) གག(missing everything except for the suffix) ག(missing everything).

In the example above, the fifth weight and Wazur are missing, this is because there is no Wazur so it’s missing.

**Significance of the Unicode values**
The input Tibetan has to be encoded with Unicode, otherwise any program relaying on this code will break or not work properly.
This work heavily depends on Unicode values. For example, གིགས་ གིངས་ གིད will be ordered before གྱ obviously, because although ི and ྱ both have the same weight, the third weight, but ི is a unicode value of OF72 while ྱ is OF74. This will always precede ི before ྱ and thus this example works like so.

**Discrepancy with Tibetan dictionary publications**
In the Newly Compiled Dictionaries (དག་ཡིག་གསར་བསྒྲིག), they put heaviest weight on prefixes(if we use the same logic above). That will make བསྒྲིགས་ order before བགག, which is questionable.

Issues:
1. Not tested will, test codes are under the way.
2. Post-suffix ར (archaic)is not detected for ordering.

**Technical details**
WordGranules.php is designed on balanced tree with max depth of 10, so it’s linear to the size of input (number of the words separated by ཚེག་ ‘’).