Lab 08 Function 2

Note: The underlined text is an input data.

1. **Write the program that sorts student data by score in descending order. The student data comes from file specified by command-line arguments.**

   The student data is in the following format

   ```
   number_of_data
   student_name1 section1 score1
   student_name2 section2 score2
   ...
   ```

   PHP provides built-in function `usort()` with the following description.

   usort — Sort an array by values using a user-defined comparison function

   ```
   bool usort ( array &$array , callable $value_compare_func )
   ```

   Parameters

   `$array`

   The input array.

   `$value_compare_func`

   The comparison function must return an integer less than, equal to, or greater than zero if the first argument is considered to be respectively less than, equal to, or greater than the second.

   Try to learn about `usort` from http://php.net/manual/en/function.usort.php by yourself.

File: `ass-01-input.txt`	**Example:** `php ass-01.php ass-01-input.txt`

`3`
`Adam 002 67.59`
`Bob 001 30.50`
`Dan 001 80.97`
	`Dan 001: 80.97`
	`Adam 002: 67.59`
	`Bob 001: 30.50`

2. Write the program that calculate average score and the summation of total score that are greater than or equal average score. The student data file and specific field come from command-line arguments.

   The student data is in the following format

```
number_of_data
student_name1 section1 chapter1_score1 chapter2_score1
student_name2 section2 chapter1_score2 chapter2_score2
...
```

   File: `ass-02-input-01.txt`

```
8
Adam 001 25.00 25.00
Alex 001 10.50 39.50
Candy 001 50.25 24.75
Alex 002 10.00 15.00
Bob 002 12.50 12.50
Candy 002 35.00 40.00
Frank 002 0.00 0.00
Susan 002 50.00 50.00
```

   Example 1: `php ass-02.php ass-02-input-01.txt`

```
Adam       001:  25.00  25.00 =  50.00
Alex       001:  10.50  39.50 =  50.00
Candy      001:  50.25  24.75 =  75.00
Alex       002:  10.00  15.00 =  25.00
Bob        002:  12.50  12.50 =  25.00
Candy      002:  35.00  40.00 =  75.00
Frank      002:   0.00   0.00 =   0.00
Susan      002:  50.00  50.00 = 100.00

Average total score :  50.00
Summation of total score greater than or equal  50.00 : 350.00
```

File: `ass-02-input-02.txt`

```
8
Adam 001 25.00 24.00
Alex 001 10.50 39.50
Candy 001 50.25 24.75
Alex 002 10.00 15.00
Bob 002 12.50 12.50
Candy 002 35.00 40.00
Frank 002 0.00 0.00
Susan 002 50.00 50.00
```

Example 2: `php ass-02.php ass-02-input-02.txt`

```
Adam       001:  25.00  24.00 =  49.00
Alex       001:  10.50  39.50 =  50.00
Candy      001:  50.25  24.75 =  75.00
Alex       002:  10.00  15.00 =  25.00
Bob        002:  12.50  12.50 =  25.00
Candy      002:  35.00  40.00 =  75.00
Frank      002:   0.00   0.00 =   0.00
Susan      002:  50.00  50.00 = 100.00

Average total score :  49.88
Summation of total score greater than or equal  49.88 : 300.00
```

3. **Write the program that sorts student data by specific field and print out average total score. The student data file and specific field come from command-line arguments. The student data contain section of each student and the optional command-line arguments can specific section of output (if it was not specified then display all).**

   The command-line arguments are in the following pattern.

   ```
   php ass-02.php filename specific_field [section]
   ```

   The student data is in the following format

   ```
   number_of_data
   student_name1 section1 chapter1_score1 chapter2_score1 chapter3_score1
   student_name2 section2 chapter1_score2 chapter2_score2 chapter3_score2
   ...
   ```

   The *specific_field* is one of the following values:

`name`	sort by name then section.
`section`	sort by section then name.
`1`	sort by chapter 1 score.
`2`	sort by chapter 2 score.
`3`	sort by chapter 3 score.
`total`	sort by total score.

   If `1`, `2`, `3` or `total` has the same score then sort by section then name.

   File: `ass-03-input.txt`

   ```
   8
   Susan 002 15.50 2.00 2.50
   Alex 002 4.50 10.50 5.00
   Candy 001 20.00 4.00 1.00
   Bob 002 10.50 5.00 10.00
   Adam 001 50.00 25.00 25.00
   Alex 001 9.50 9.00 1.50
   Candy 002 25.00 50.00 25.00
   Frank 002 10.00 4.50 10.50
   ```

Example 1: `php ass-03.php ass-03-input.txt name`

```
Adam       001:  50.00  25.00  25.00 = 100.00
Alex       001:   9.50   9.00   1.50 =  20.00
Alex       002:   4.50  10.50   5.00 =  20.00
Bob        002:  10.50   5.00  10.00 =  25.50
Candy      001:  20.00   4.00   1.00 =  25.00
Candy      002:  25.00  50.00  25.00 = 100.00
Frank      002:  10.00   4.50  10.50 =  25.00
Susan      002:  15.50   2.00   2.50 =  20.00

Average total score =  41.94
```

Example 2: `php ass-03.php ass-03-input.txt section`

```
Adam       001:  50.00  25.00  25.00 = 100.00
Alex       001:   9.50   9.00   1.50 =  20.00
Candy      001:  20.00   4.00   1.00 =  25.00
Alex       002:   4.50  10.50   5.00 =  20.00
Bob        002:  10.50   5.00  10.00 =  25.50
Candy      002:  25.00  50.00  25.00 = 100.00
Frank      002:  10.00   4.50  10.50 =  25.00
Susan      002:  15.50   2.00   2.50 =  20.00

Average total score =  41.94
```

Example 3: `php ass-03.php ass-03-input.txt 1`

```
Alex       002:   4.50  10.50   5.00 =  20.00
Alex       001:   9.50   9.00   1.50 =  20.00
Frank      002:  10.00   4.50  10.50 =  25.00
Bob        002:  10.50   5.00  10.00 =  25.50
Susan      002:  15.50   2.00   2.50 =  20.00
Candy      001:  20.00   4.00   1.00 =  25.00
Candy      002:  25.00  50.00  25.00 = 100.00
Adam       001:  50.00  25.00  25.00 = 100.00

Average total score =  41.94
```

Example 4: `php ass-03.php ass-03-input.txt 3`

```
Candy      001:  20.00   4.00   1.00 =  25.00
Alex       001:   9.50   9.00   1.50 =  20.00
Susan      002:  15.50   2.00   2.50 =  20.00
Alex       002:   4.50  10.50   5.00 =  20.00
Bob        002:  10.50   5.00  10.00 =  25.50
Frank      002:  10.00   4.50  10.50 =  25.00
Adam       001:  50.00  25.00  25.00 = 100.00
Candy      002:  25.00  50.00  25.00 = 100.00

Average total score =  41.94
```

Example 5: `php ass-03.php ass-03-input.txt total`

```
Alex       001:   9.50   9.00   1.50 =  20.00
Alex       002:   4.50  10.50   5.00 =  20.00
Susan      002:  15.50   2.00   2.50 =  20.00
Candy      001:  20.00   4.00   1.00 =  25.00
Frank      002:  10.00   4.50  10.50 =  25.00
Bob        002:  10.50   5.00  10.00 =  25.50
Adam       001:  50.00  25.00  25.00 = 100.00
Candy      002:  25.00  50.00  25.00 = 100.00

Average total score =  41.94
```

Example 6: `php ass-03.php ass-03-input.txt 3 001`

```
Candy       001:  20.00   4.00   1.00 =  25.00
Alex        001:   9.50   9.00   1.50 =  20.00
Adam        001:  50.00  25.00  25.00 = 100.00

Average total score =  48.33
```

Example 7: `php ass-03.php ass-03-input.txt total 001`

```
Alex        001:   9.50   9.00   1.50 =  20.00
Candy       001:  20.00   4.00   1.00 =  25.00
Adam        001:  50.00  25.00  25.00 = 100.00

Average total score =  48.33
```

Example 8: `php ass-03.php ass-03-input.txt total 002`

```
Alex        002:   4.50  10.50   5.00 =  20.00
Susan       002:  15.50   2.00   2.50 =  20.00
Frank       002:  10.00   4.50  10.50 =  25.00
Bob         002:  10.50   5.00  10.00 =  25.50
Candy       002:  25.00  50.00  25.00 = 100.00

Average total score =  38.10
```