Task 1

Preamble

Marvel Studios' latest film, Deadpool & Wolverine, has garnered unprecedented hype. Ryan Reynolds, reprising his role as Deadpool, and Hugh Jackman as Wolverine, are set to create cinematic history with the first collaboration between the two characters on screen (if you know, you know). However, behind the scenes, there's a high-stakes bet between Marvel Studios President Kevin Feige and Ryan Reynolds.

Feige, known for his strategic mind and his ability to predict box office hits, believes he can assemble a cast of cameos that will maximise the movie's revenue without surpassing a certain budget. Reynolds, always up for a challenge, bets that if the total estimated revenue (in millions) from these cameos exceeds a certain limit, he should get to direct the next Avengers movie, titled Avengers: Doomsday.

Your mission is to help Kevin Feige win this bet. You need to write a Java program that reads a list of potential cameos and their associated revenue contributions, then selects the optimal combination of cameos that maximises the total number of cameos without exceeding the given revenue limit.

Note: There are no spoilers in this task.

Task

The target score will be provided as a command-line argument, and you will continuously receive input in the form of character-score pairs via standard input (stdin). Each pair will consist of a character's name and their associated revenue (in millions of dollars). You must process this input, select the optimal set of characters, and print the results in the described format.

Requirements

Input Handling

The program will read input from **standard input** (`stdin`). Each line of input will contain one or more character-score pairs, separated by spaces. For example:

```
Deadpool 50 Wolverine 60 ProfessorX 40
```

Each character name is a single word, and the score is an integer or float. If a particular character is especially hated, it is allowable for them to have a score of 0, but their score may not be negative. Any number of characters can be entered in a single line (0 or more), and you should not stop taking input until the string `EOF` is entered as input on a line on its own.

Output Handling

The program must print the selected characters in alphabetical order. The output format should be:

```
Selected characters: [Character1, Character2, ..., CharacterN]
Total expected revenue: <X> million dollars
```

Characters should be listed in alphabetical order.

If no valid characters can be selected without exceeding the target score, print:

```
No characters selected.
```

Error Handling

If the input format is incorrect, the program should **ignore the entire line of input**, print the following error message and continue to process any remaining valid input:

```
Input '<user_input>' is not valid. Make sure your input is valid.
```

Constraints:

- You are not allowed to use custom classes or constructors.
- The entire program must be contained within a single file.
- The target score will be passed as a command-line argument (`argv`).

Example Execution

Example 1

Suppose the target score is 100 million dollars. We would run the program as follows:

```
javac DeadpoolAndWolverine.java
java DeadpoolAndWolverine 100
```

We could then enter the following input:

```
Deadpool 50 Wolverine 60 Professor_X 40
IronMan 30 Captain_America 40
EOF
```

The program should output:

```
Selected characters: [Captain_America, IronMan]
Total expected revenue: 70 million dollars
```

Example 2

Let's consider a more detailed example to illustrate all aspects of the task. Here, we'll show you the entire stream you would see in your terminal (including all input, output and commands).

```
javac DeadpoolAndWolverine.java
java DeadpoolAndWolverine 150

Deadpool 50 Wolverine 60
Iron_Man 30 Captain_America 40
Thor 20 Loki 15
Hulk 55 Spider-Man 45
EOF

Selected characters: [Captain_America, Iron_Man, Loki, Spider-Man, Thor]
Total expected revenue: 150 million dollars
```

Other Details

Output

The selected characters are printed in alphabetical order, along with the total expected revenue as described above.

Ensure that your implementation follows these guidelines and handles all edge cases as specified. Good luck!

Remember: Don't overthink things! The easiest way to get the maximum number of characters is to order them in increasing value and continuously choose the lowest-value character until you reach your target. Remember that your goal is to include as many characters as possible while minimising your total below the budget.

Frequenly Asked Questions

What if two characters have the same value?

In the case that two characters have the same value, you should choose the character that appears first alphabetically.

What about errors that could be made that aren't described in the specifications?

Any possible erroneous circumstances that a user may encounter that haven't been described in the specifications do not need to be considered and will not be tested in the test cases.

Will all the assessed tasks be Marvel-related?

Maybe? Possibly? You'll find out soon ☺