Get Set!

This is an exercise with sets. Simply devising the data structures and algorithms to do it is sufficient. Concern about time efficiency is not. The idea is to build and manipulate sets. Submit your solution under the file name *GetSet* with the appropriate language extension.

Input

The input consists of a sequence of at most 99 commands, one per line, ending with a line containly only the word "stop" (and perhaps a new line character). Each command line begins with one of the words in the following list. All commands in the input will be legal and executable. The stack is initially empty. The sequence may end with sets still in the stack.

set - The positive integers that follow define a new set. Add it to the stack of previously defined sets. There will be at most 10 integers for the new set.
union - Replace the top two sets on the stack with their union.
intersect - Replace the top two sets on the stack with their intersection.
diff - Replace the top two sets on the stack with their set difference (the topmost minus the next to the top).
symdiff - Replace the top two sets on the stack with their symmetric difference.
sub - Move to the top of the stack the highest set that is a proper subset of the top set.
super - Move to the top of the stack the highest set that is a proper superset of the top set.
dup - Duplicate the set at the top of the stack.
swap - Swap the top two sets on the stack.
rot - Rotate the top three sets on the stack, bringing to the top the third set down.
print - Pop and print the set at the top of the stack.

Output

Output occurs with each "print" command. Enumerate and format as in the example. Spacing is important. Elements should always be in sorted order.

Sample Input

```
set 2 6 4 3 1
dup
set 1 3 5 7
dup
rot
union
diff
super
print
print
stop
```

Sample Output

```
1. [1 2 3 4 6]
2. [2 4 6]
```