CS 158/159 Homework 7

This assignment is worth 15 points and will be **due Monday April 23, 2018 at 11:00pm**. All assignment deadlines are firm and the ability to submit your assignment will be disabled after the deadline elapses. No late work will be accepted. You are encouraged to start this assignment as soon as possible so that you have ample time to seek assistance should you experience difficulties completing or submitting this assignment.

This programming assignment does not have a single solution, and the assignment you submit must be your own original work. **Collaboration with other students is not permitted on homework assignments.** Any submission may be processed with comparison software and the results will be used to detect unacceptable collaboration between individuals. If you need assistance, you should only consult course staff regarding your program.

Your program must adhere to the course programming standards (available in the course packet and in Blackboard). Please review this document before submitting your assignment, because failure to adhere to it will result in a reduction in points. Your program must include the designated program header (~cs15x/student/hdrProg) at the top of the program (which can be inserted in `vi` using the `hp` shortcut while in command mode). The header must include an appropriate description of your program and must contain your official Purdue career account e-mail address. Also note that course standards prohibit the use of advanced programming concepts not yet introduced in the course, unless otherwise specified.

Each of the example executions provided below represents a single execution of the program. Your program must accept input and produce output **exactly** as demonstrated in the example executions. Your program will be tested with the data seen in the examples below and an unknown number of additional tests making use of reasonable data. Do not include any example outputs with your submission.

A single program file (with the `.c` extension) must be submitted electronically via the `guru` server. An example submission was conducted during the first week in lab00. Any attempts to submit via another method will be denied consideration. You may make multiple submissions before the deadline, but only the last attempt is retained and graded. All previous submissions will be over-written and cannot be recovered. The submission script will reject the submission of any file that does not compile. A program must compile to be considered for partial credit. You should always check the confirmation e-mail you receive after a submission to verify that you have submitted the correct file, to the correct assignment, and to the correct lab section. If you have a concern regarding how to submit work, please visit course staff prior to the assignment deadline.

Problem: Each coin in a collection consists of a value (denomination) and a date in which it was minted. A collector will input up to 50 coins and would like to know if for any date or value whether their set is complete. A `-1` value is entered to indicate that all of the data has been entered. This is not done, however, if the maximum number of coins (50) has been entered.

What constitutes being "complete" is dependent on the full collection given as input. For example, one user may enter the following coins [denomination, date]: [5, 2010], [10, 2010], [1, 2010], [5, 2015], [10, 2015]. In this five-coin collection there are three different denominations (1, 5, 10) and two different dates (2010, 2015). A complete set would be all three denominations for a given year or all years for a given denomination. In this example set there is a complete collection of the 5 and 10 denominations, and the set is complete for the year 2010. The 1 denomination coin is missing from 2015.

Example Execution #1 (displays those dates present in the set at least once):

```
Enter data now -> 10 2015 5 2010 5 2015 10 2010 1 2010 -1

1. Display unique dates.
2. Display unique denominations.
3. Display incomplete set data for dates.
4. Display incomplete set data for coins.
Enter option for data set (size 5) -> 1

Unique dates in set: 2015 2010
```

Example Execution #2 (displays those denominations present in the set at least once):

```
Enter data now -> 10 2015 5 2010 5 2015 10 2010 1 2010 -1

1. Display unique dates.
2. Display unique denominations.
3. Display incomplete set data for dates.
4. Display incomplete set data for coins.
Enter option for data set (size 5) -> 2

Unique denominations in set: 10 5 1
```

Example Execution #3:

```
Enter data now -> 10 2015 5 2010 5 2015 10 2010 1 2010 -1

1. Display unique dates.
2. Display unique denominations.
3. Display incomplete set data for dates.
4. Display incomplete set data for coins.
Enter option for data set (size 5) -> 3

Incomplete sets for the following dates: 2015
```

- The date 2015 is missing the 1 denomination that is present in this set.

Example Execution #4:

```
Enter data now -> 10 2015 5 2010 5 2015 10 2010 1 2010 -1

1. Display unique dates.
2. Display unique denominations.
3. Display incomplete set data for dates.
4. Display incomplete set data for coins.
Enter option for data set (size 5) -> 4

Incomplete sets for the following coins: 1
```

- The 1 denomination is missing for the date 2015 that is present in this set.

Example Execution #5 (dates displayed in descending order):

```
Enter data now -> 10 1900 5 2000 10 1900 5 2005 1 2000 1 1900 10 1800 1 1800 1 1800
1 2010 10 2010 10 2010 5 2005 -1

1. Display unique dates.
2. Display unique denominations.
3. Display incomplete set data for dates.
4. Display incomplete set data for coins.
Enter option for data set (size 13) -> 1

Unique dates in set: 2010 2005 2000 1900 1800
```

Example Execution #6 (denominations displayed in descending order):

```
Enter data now -> 10 1960 75 1960 100 1960 200 1960 1 1960 3 1960 50 1960 25 1960 -1

1. Display unique dates.
2. Display unique denominations.
3. Display incomplete set data for dates.
4. Display incomplete set data for coins.
Enter option for data set (size 8) -> 2

Unique denominations in set: 200 100 75 50 25 10 3 1
```

Example Execution #7:

```
Enter data now -> 1 2000 5 2000 25 2000 1 2010 25 2010 5 2010 25 2015 5 2015 1 2015
5 1995 1 1995 25 1995 -1

1. Display unique dates.
2. Display unique denominations.
3. Display incomplete set data for dates.
4. Display incomplete set data for coins.
Enter option for data set (size 12) -> 3

There are no incomplete sets of dates.
```

Example Execution #8:

```
Enter data now -> 1 2000 5 2000 25 2000 1 2010 25 2010 5 2010 25 2015 5 2015 1 2015
5 1995 1 1995 25 1995 -1

1. Display unique dates.
2. Display unique denominations.
3. Display incomplete set data for dates.
4. Display incomplete set data for coins.
Enter option for data set (size 12) -> 4

There are no incomplete sets of coins.
```

Example Execution #9 (input validation requirements demonstrated):

```
Enter data now -> 5 2018 -1

1. Display unique dates.
2. Display unique denominations.
3. Display incomplete set data for dates.
4. Display incomplete set data for coins.
Enter option for data set (size 1) -> 5

Error! Select a valid option from the menu above!

1. Display unique dates.
2. Display unique denominations.
3. Display incomplete set data for dates.
4. Display incomplete set data for coins.
Enter option for data set (size 1) -> 0

Error! Select a valid option from the menu above!

1. Display unique dates.
2. Display unique denominations.
3. Display incomplete set data for dates.
4. Display incomplete set data for coins.
Enter option for data set (size 1) -> 2

Unique denominations in set: 5
```

Example Execution #10 (coins may be present more than once in the set):

```
Enter data now -> 1 2010 1 2012 5 2010 5 2012 5 2016 1 2010 10 2010 25 2010 10 2012
10 2016 25 2012 25 2016 1 2012 5 2012 25 2010 25 2016 10 2012 10 2010 10 2012 10
2016 1 2010 1 2012 -1

1. Display unique dates.
2. Display unique denominations.
3. Display incomplete set data for dates.
4. Display incomplete set data for coins.
Enter option for data set (size 22) -> 4

Incomplete sets for the following coins: 1
```

Example Execution #11:

```
Enter data now -> 1 2010 1 2012 5 2010 5 2012 5 2016 1 2010 10 2010 25 2010 10 2012
10 2016 25 2012 25 2016 1 2012 5 2012 25 2010 25 2016 10 2012 10 2010 10 2012 10
2016 1 2010 1 2012 -1

1. Display unique dates.
2. Display unique denominations.
3. Display incomplete set data for dates.
4. Display incomplete set data for coins.
Enter option for data set (size 22) -> 3

Incomplete sets for the following dates: 2016
```

Example Execution #12 (when 50 coins are present a -1 is not needed to terminate input):

```
Enter data now -> 1 1920 5 1920 10 1920 25 1920 50 1920 1 1919 5 1919 10 1919 25
1919 50 1919 50 1917 25 1917 10 1917 5 1917 1 1917 1 1918 5 1918 10 1918 25 1918 50
1919 50 1917 25 1917 10 1917 5 1917 1 1917 1 1920 5 1920 10 1920 25 1920 50 1920 1
1919 5 1919 10 1919 25 1919 50 1919 50 1917 25 1917 10 1917 5 1917 1 1917 1 1918 5
1918 10 1918 25 1918 50 1918 50 1917 25 1917 10 1917 5 1917 1 1917

1. Display unique dates.
2. Display unique denominations.
3. Display incomplete set data for dates.
4. Display incomplete set data for coins.
Enter option for data set (size 50) -> 2

Unique denominations in set: 50 25 10 5 1
```

Example Execution #13 (demonstrates that there can be more than one incomplete denomination):

```
Enter data now -> 1 2018 10 2017 5 2016 50 2015 25 2014 -1

1. Display unique dates.
2. Display unique denominations.
3. Display incomplete set data for dates.
4. Display incomplete set data for coins.
Enter option for data set (size 5) -> 4

Incomplete sets for the following coins: 50 25 10 5 1
```

Example Execution #14 (demonstrates that there can be more than one incomplete date):

```
Enter data now -> 1 2018 10 2017 10 2018 5 2016 50 2015 5 2018 25 2014 25 2018 50
2018 -1

1. Display unique dates.
2. Display unique denominations.
3. Display incomplete set data for dates.
4. Display incomplete set data for coins.
Enter option for data set (size 9) -> 3

Incomplete sets for the following dates: 2017 2016 2015 2014
```

Additional Notes:

- Only option input from the menu requires validation (see example #9).
- All data for denominations and dates will be integers and positive.
- Do not assume the only denominations are those current denominations in your world currency of choice.
- There will be at least one coin in every collection.
- The terminal -1 value is only needed when the total number of coins is less than 50. Hint: 49 is less than 50.
- Course standards prohibit the use of programming concepts not yet introduced in lecture. For this assignment you can only consider material in the first 8 chapters of the book, notes, and lectures. Use of advanced programming constructs beyond this material would result in a loss of points.