Project [2]: Logical Expressions and Branching Statements

Due Thursday, 9/20/2018, 11:59 pm

Project Goals

The goals of this project are to:

1. Get students familiar with evaluating logical expressions
2. Get students familiar with `if - else` statements
3. Get students familiar with `if - else if - else` statements

Important Notes:

1. **Formatting:** Make sure that you follow the precise recommendations for the output content and formatting: for example, do not change the text of the problem from “`Enter volumes and price per volume for collection 1: `” to "`Enter volume, price: `". Your assignment will be auto-graded and any change in formatting will result in a loss in the grade.
2. **Comments:** Header comments are required on all files and recommended for the rest of the program. Points will be deducted if no header comments are included.

Problem 1

Write a program that asks the user to enter information about three book collections. Each collection has a number of volumes and a price per volume (same for each volume in the collection). The program should ask how many volumes and what is the price per volume for each collection, as well as a budget available for a buyer named Bob. All values entered should be integers. The program should evaluate a set of logical expressions and print out specific messages based on the truth value of those expressions. The expressions are as follows:

(1) *Bob has money*.
If the condition is true (budget is greater than 0) the program should print:

```
(1) Bob has some money to buy collections.
```

If the condition is false, the program should print:

```
(1) Bob does not have money to buy anything.
```

(2) *At least two collections are more expensive than Bob’s Budget.*
If the condition is true the program should print:

```
(3) At least two collections are more expensive than Bob’s budget.
```

If the condition is false, the program should print:

```
(3) At least two collections are cheaper than or equal to Bob’s budget.
```

(3) *All the collections cost the same, only two collections cost the same, or no collections cost the same.*
If all the collections cost the same, the program should print:

```
(4) All the collections cost the same.
```

If only two collections cost the same, the program should print:

```
(4) Only two collections cost the same amount of money.
```

If none of the collections cost the same, the program should print:

```
(4) No collections have the same price.
```

(4) *Only one collection is cheaper than or equal to Bob’s budget.*
If the condition is true, the program should print:

```
(4) Only one collection is cheaper than or equal to Bob’s budget.
```

If the condition is false, the program should print:

```
(4) More than one collection is cheaper than or equal to Bob’s budget or they are
all more expensive.
```

(5) *The maximum number of collections that Bob can buy.*
If Bob can buy all the collections, the program should print:

```
(5) Bob can buy all three collections.
```

If Bob can only buy two of the collections, the program should print:

```
(5) Bob can only buy two of the collections.
```

If Bob can only buy one of the collections, the program should print:

```
(5) Bob can only buy one collection.
```

If Bob can’t buy any collection, the program should print:

```
(5) Bob cannot buy any collection.
```

The program should function as follows (items underlined are to be entered by the user):

```
Enter volumes and price per volume for collection 1: 2 20
Enter volumes and price per volume for collection 2: 3 30
Enter volumes and price per volume for collection 3: 4 50
Enter Bob’s budget: 100
(1) Bob has some money to buy collections
(2) At least two collections are cheaper than or equal to Bob's budget.
(3) No collections have the same price.
```

```
(4) More than one collection is cheaper than or equal to Bob's budget or they are all more
expensive
(5) Bob can only buy one collection.
```

Save your program as `collections.c`

Challenge for problem 1 (10 extra credit points):
Your program should also check the following conditions (you can add them at the end of the previous 5 in the original program):

(6) *Bob has enough money to buy any one of the three collections.*
If the condition is true the program should print:

```
(6) Bob has enough money to buy any one of the three collections.
```

If the condition is false, the program should print:

```
(6) Bob does not have enough money to buy any one of the three collections.
```

(7) *Bob does not have enough money to buy any collection*.
If the condition is true, the program should print:

```
(7) Bob does not have enough money to buy any collection.
```

If the condition is false, the program should print:

```
(7) Bob can buy at least one collection.
```

The program should function as follows (items underlined are to be entered by the user):

```
Enter volumes and price per volume for collection 1: 2 20
Enter volumes and price per volume for collection 2: 3 30
Enter volumes and price per volume for collection 3: 4 50
Enter Bob’s budget: 100
(1) Bob has some money to buy collections
(2) At least two collections are cheaper than or equal to Bob's budget.
(3) No collections have the same price.
(4) More than one collection is cheaper than or equal to Bob's budget or they are all more
expensive
(5) Bob can only buy one collection.
(6) Bob does not have enough money to buy any one of the three collections.
(7) Bob can buy at least one collection.
```

Save your challenge separately as `collections_c.c`

Grading Rubric

Grading will be done for each problem as follows:

Correctly-named file	5%
Header comment	2%
Program compiles	5%
Correctly-reading data from terminal	28%
Correct result printed	60%

Submission details

To submit your project, you will have to use the submission script. You do this by either:

1. Working on an ECC machine
2. Working on the provided VMware
3. Secure Copying your files (See Mac Support for information)

To Submit your project:

- Have a directory called "project2"
- Save your *.c files in that directory
- To submit: **(don't type the '>' symbols)**

```
> cd project2
> submit
```

The submission script copies all files in the current directory to our directory. You may submit as many times as you like before the deadline, we only keep the last submission.

Academic Honesty

Academic dishonesty is against university as well as the system community standards. Academic dishonesty includes, but is not limited to, the following:

Plagiarism: defined as submitting the language, ideas, thoughts or work of another as one's own; or assisting in the act of plagiarism by allowing one's work to be used in this fashion.

Cheating: defined as (1) obtaining or providing unauthorized information during an examination through verbal, visual or unauthorized use of books, notes, text and other materials; (2) obtaining or providing information concerning all or part of an examination prior to that examination; (3) taking an examination for another student, or arranging for another person to take an exam in one's place; (4) altering or changing test answers after submittal for grading, grades after grades have been awarded, or other academic records once these are official.

Cheating, plagiarism or otherwise obtaining grades under false pretenses" constitute academic dishonesty according to the code of this university. Academic dishonesty will not be tolerated and penalties can include cancelling a student’s enrolment without a grade, giving an F for the course, or for the assignment. For more details, see the University of Nevada, Reno General Catalog.