**Assignment #7**

**Introduction to C Programming – COP 3223**

## Objective

1. To learn how to write functions given specifications
2. To learn how to the use pass by value and pass by reference variables
3. To practice planning and implementing complex programs.

## Introduction: Programmers for a Better Tomorrow

Programmers for a Better Tomorrow is an organization dedicated to helping charities, medical societies, and scholarship organizations manage various tasks so that they can focus on making the world a better place! They have asked you and your classmates to help them develop some new programs to benefit their organizations.

## Problem: Charity Gala (gala.c)

Many charities support good causes, but one of the difficulties each of them has is organizing their fundraising events. You've decided that you'd like to donate your skills to create a program that organizes the typical activities at a fund-raising gala. In particular, your program will help manage the following:

1) Ticket sales
2) Silent Auction
3) Raffle

Your program will log the number of tickets sold both in advance and at the event, a silent auction for donated items, and a raffle for other donated items.

## Program Details

*Ticket Sales Details*

You will sell tickets in advance and at the door. Prices for buying in advance and at the door will be given. Also, the total number of tickets sold in advance will be given. Each guest will have a unique number. If there are *n* tickets sold in advance, then these guests will be numbered 0 through *n-1*. As the event starts, requests to buy tickets at the door may be made and these guests will be numbered sequentially, starting at the lowest unassigned number. The maximum number of guests will be 1000.

*Silent Auction Details*

The silent auction will have up to 1000 items. At any time, users can bid on any item, so long as the new bid exceeds the previous bid. At the point in time when the auction is closed, the items with at least one bid are given to the users who have placed the highest bid on that item. Your program must always keep track of all of the best bids on items so that no matter when the auction closes, you'll have all the data for who has won all of the items. Any item without a bid goes to no one.

*Raffle Details*
There will be a given number for the total number of raffle tickets sold, that will be 1000 or fewer. Guests can buy raffle tickets for a set price of $2.00. There will be a number of prizes awarded from the raffle (not to exceed 100) after the raffle has finished. For each raffle ticket, you'll have to keep track of which guest has it. The raffle tickets will be numbered starting at *0* through the number of tickets minus one. After the raffle finishes, you will be given the raffle numbers pulled for the winners of each prize. You will be guaranteed that these numbers correspond to numbers that were previously handed out during the raffle. You must determine who wins each prize.

Information for each event is provided from a file. You will take in input from the file and produce output for each event that occurs, in order.

A skeleton of the solution for this assignment is posted on the webcourse. You must fill in the functions that are currently empty. After you write each function, you should test it before moving on. The main function should not be modified for the final submission (you may modify it during testing, as long as you return it to its initial form).

Descriptions of each function are given in the skeleton along with the function Pre- and Post-conditions. The output sample on the webcourse shows the wording you should use and how the program should run when completed. Points are allotted for following the precise wording shown.

**Input Specification**
The first line of the file contains the following three values, separated by spaces:

> Cost of the presales tickets (in dollars), Cost of the tickets at the door (in dollars), and the number of presale tickets. The first two values will be positive real numbers to two decimal places and the last will be a positive integer.

The second line of the file will contain one positive integer representing the number of auction items.

The third line of the file will contain the two following positive integers pertaining to the raffle: the number of raffle tickets available and the number of raffle prizes.

The fourth line of the file will contain a single positive integer, *numEvents*, representing the number of events that occur at the fundraising event. These events are split into two groups: actions by guests at the ball and awards given (raffle, auction, person, totalrevenue). All of the actions precede all of the awards.

The following *numEvents* lines will contain information about each event that occurs, with one event or award described for each line.

You will produce exactly one line of output for each event described. Here are the formats of each event that could occur:

If a patron buys a ticket at the door, a command will be on a line by itself:

BUY TICKET k

where k is a positive integer indicating the number of tickets bought at the door. These guests will be numbered as previously mentioned. You are guaranteed that the total number of tickets bought, including presales, will not exceed 1000.

If a patron makes a bid in the silent auction, a command of the following form will be issued:

BIDITEM k p d

where k represents the item number for which the bid is being placed, p represents the number of the person placing the bid, and d represents the dollar amount the person has bid. This last value is a positive real number to two decimal places while the others are non-negative integers within the appropriate ranges.

The following command closes the auction. Any bids made after this command is executed are ignored.

CLOSEAUCTION

If a guest desires to buy raffle tickets, a command with the following format will be used:

BUY RAFFLE k p

This command lets person p buy exactly k raffle tickets. The raffle tickets are numbered as previously described, so long as the tickets are still available. If they aren't available, the person gets whatever tickets remain.

The last several commands in the file will deal with awarding prizes. These commands will take the following forms.

A query about a raffle item will be of the following form:

AWARD RAFFLE i t

where i is the raffle item being given out and t is the number of the winning ticket of that raffle item. Your program will have to respond with the winner of this item. You will be guaranteed that this query will appear EXACTLY once for each raffle prize and that all queries of this form will appear AFTER the last raffle ticket is sold. Furthermore, you are guaranteed that the raffle ticket number was purchased by an individual and that the raffle prize number is valid. Finally, a single raffle ticket will correspond to at most one raffle prize.

A status query about a silent auction item will be of the following form:

AWARD AUCTION i

where i represents the silent auction item being queried. Your program will have to respond with the winner of this item and the amount they paid for it OR that there were no bids for that item.

The final status query in the file will be the following line:

TOTAL REVENUE

**Output Specification**

For the input command of the form:

BUY TICKET k

output a single line of the form:

SOLD TICKETS a - b

where a is the starting number of the tickets sold and b is the ending number (inclusive) of the tickets sold. If k is 1, then a and b will be equal. (Note: For the very first command of this type, the value of a will equal the number of presold tickets, since the presale tickets are numbered from 0 to the total number of presale tickets minus 1.)

For the input command of the form:

BIDITEM k p d

output a single line with one of the following forms:

BIDITEM k ACCEPTED for PERSON p at d DOLLARS
BIDITEM k REJECTED for PERSON p at d DOLLARS

The two reasons to reject a bid are if the new bid is not higher than the previous bid OR if the auction has been closed already.

For the CLOSEAUCTION command, simply print a single line with the exact same output:

CLOSEAUCTION

For the input command

BUY RAFFLE k p

output a single line of one of the two following forms:

RAFFLE TICKETS a - b given to PERSON p
NO RAFFLE TICKETS given to PERSON p

where a represents the first raffle ticket number issued and b represents the last raffle ticket number issued to person p. If there are no more raffle tickets left with the command is issued, the latter format is used. Note that the former format may be used if the person asks for a certain number of raffle tickets, but gets fewer since she bought all that were left.

For the input command

AWARD RAFFLE i t

output a single line of the following format:

RAFFLE i WON BY PERSON p

where i is the raffle item in question (in the query) and p is the number of the guest who won the item.

For the input command

AWARD AUCTION i

output a single line of the following format:

AUCTION ITEM i WON BY PERSON p for $d.dd

where i is the number of the auction item, p is the person who won the item and d.dd is the number of dollars he paid for it, to two decimal places. Or, if there are no bids for the item, output a single line of the following format:

NO BIDS FOR AUCTION ITEM %d

For the input command

TOTALREVENUE

output a single line with the following format:

TOTALREVENUE is $d.dd

where d.dd is the total amount gained by the event via ticket sales, auction sales, and raffle ticket sales. (We are assuming that all items were donated so that there was no cost associated with the event.)

**Sample Input/Output**
Sample input and output files will be provided on the webcourse.

**Deliverables**
A single source file named *gala.c* turned in through WebCourses.

**Restrictions**

Although you may use other compilers, your program must compile in gcc and run in Code::Blocks. Your program should include a header comment with the following information: your name, course number, section number, assignment title, and date. Make sure you include comments throughout your code describing the major steps in solving the problem.

Make sure to use good programming style, including use of appropriate constants, good variable names and good use of white space.

**Grading Details**

Your programs will be graded upon the following criteria:

1) Your correctness

2) Your programming style and use of white space. Even if you have a plan and your program works perfectly, if your programming style is poor or your use of white space is poor, you could get 10% or 15% deducted from your grade.

3) Compatibility – You must submit C source files that can be compiled and executed in a standard C Development Environment. If your program does not compile, you will get a sizable deduction from your grade.