EE231002 Introduction to Programming

Lab08. Blackjack

Due: Apr. 26, 2014

Blackjack is a popular card game. In this game, a player is dealt two cards initially. The number of points can then be calculated as follows. The cards **2** to **10** have the face values, face cards (**J**, **Q**, and **K**) have 10 points, and an **A** can have either 1 or 11 points. Depending on the total points, the player can make a decision to ask for more card (hit) or not (stand). As long as the number of points does not exceed 21, the play can continue to request for more cards. The objective of the game is to get more points than the dealer without going over 21 points.

In this lab, you will practice using the `rand`() function to find the some probabilities of playing Blackjack game. In particular, you need to do two things:

1. Create a table that lists the expected value of the total number of points after the first hit, and the percent of busting.
2. Find out what is the probability of getting 21 point.

Example program output is given below.

```
Points  E(hit)  % Busted
   2      9.29         0
   3     xx.xx     xx.xx
   4     xx.xx     xx.xx
   5     xx.xx     xx.xx
   6     xx.xx     xx.xx
   7     xx.xx     xx.xx
   8     xx.xx     xx.xx
   9     xx.xx     xx.xx
  10     xx.xx     xx.xx
  11     xx.xx     xx.xx
  12     xx.xx     xx.xx
  13     xx.xx     xx.xx
  14     xx.xx     xx.xx
  15     xx.xx     xx.xx
  16     xx.xx     xx.xx
  17     xx.xx     xx.xx
  18     xx.xx     xx.xx
  19     xx.xx     xx.xx
  20     26.48     92.33

Probability of getting 21 points is xx.xx%
```

The first part of the output is a table. The first column of the table is the number of points at hand. The second column is the expected value of points after one hit (adding one card). And the third column is the percentage of busting (number of points going over 21). The last line of the output is the probability of getting 21 points. In order to get 21 points, the player is always *hit* unless it is 21 or busted.

The Blackjack game is usually played with 2 or 4 decks of cards. A deck of cards has 52 cards with 4 different suits and each suit has 13 cards. In this lab, we assume the game is played using a large number of decks and thus, the cards **A** to **K** all have equal probability whenever is dealt. You can use the following expression to draw a card:

```
k = rand() % 13 + 1;
```

The variable `k` has the value of the drawn card.

In practice, one need to play the game a large number of times to get an accurate expected value. Thus, each row of the table should be the results of at least 10000 experiments. For example, for the first row we assume there are two points at hand and request a new card. The total number of points can then be calculated. This is one experiment. Repeat this experiment 10000 times, the sum of all the points of each experiment divides 10000 is then the expected value. The percentage of busted is simply the number of experiments that the total number of points going over 21 points. In your program please define the number of experiment as a macro `N`, `N`$\geq$ 10000.

In calculating the probability of getting 21 points, the process is similar. Two cards are dealt initially, if the sum is 21 then we have a success. Otherwise, a hit is requested until 21 points are obtained or busted. The latter case is a failure. The probability is simple the number of success divided by total number of experiments.

In this game, the card **A** can be counted as 11 or 1 point. It is usually taken as an 11 if it will not result in busting, otherwise it is treated as a 1.

Notes.

1. Create a directory **`lab08`** and use it as the working directory.
2. Name your program source file as **`lab08.c`**.
3. The first few lines of your program should be comments as the following.

   ```
   /* EE231002 Lab08. Blackjack
      ID, Name
      Date:
   */
   ```

4. After you finish verifying your program, you can submit your source code by

   ```
   $ ~ee231002/bin/submit lab08 lab08.c
   ```

   If you see a "submitted successfully" message, then you are done. In case you want to check which file and at what time you submitted your labs, you can type in the following command:

   ```
   $ ~ee231002/bin/subrec
   ```

   It will show the last few submission records.

5. You should try to write the program as efficient as possible. The format of your program should be compact and easy to understand. These are part of the grading criteria.