Basic Algorithms (CS 301, Section 1), Professor Yap – Fall 2023

HOMEWORK

September 7, 2023

Homework 0: Due: 11pm Fri Sep 15, 2023

- Carefully read the Class Integrity Policy in our Class Wiki.
- Go to our Homework Page for general information about Homework.
- OVERVIEW: This is a Programming Homework. The goal is to get you up to speed on your programming environment. You need to install

  (1) a Terminal program,
  (2) a text editor (e.g., `gvim`), and
  (3) a `make` program (e.g., `gmake` is also good).
  (4) a Java Development Kit (JDK).

  The JDK provides the Java compiler `javac` and interpreter `java`. If you have any Java IDE like `Eclipse`, it would already have a JDK – there is no need to re-install another JDK. Just find out how to access its `javac/java` The TAs or tutor can help.
- Submit your answers to Gradescope (see classWiki link)

PART A: Terminal and Make Program

Your answers to PART A must be uploaded separately to GradeScope.

Question A.1 (10 Points, Terminal) Please do the following actions in a Terminal:

- Unzip `hw0-yap.zip` in some folder.
- Do `cd` ("change directory") to that folder.
- Do `pwd` ("path of current directory") to show the current path.
- Do `ls` ("list files") to show folder contents.
  E.g., you see a `Makefile` among other files of handout `hw0-yap.zip`
- Get a screenshot of your terminal, to be submitted as the file `screenshot1.jpg`

Question A.2 (10 Points, make program) Continuing from the previous question. Make sure that you can invoke the `make` program from your Terminal by typing "`make`".

- Type `make p=Hello` to run `Makefile`
- Type `make p=Hello r`
- Get a screenshot of your terminal, to be submitted as `screenshot2.jpg`
- Type `make p=Hello r0`
- Type `make p=Hello r1`
- Get a screenshot of your terminal, to be submitted as `screenshot3.jpg`

Question A.3 (10 Points, Understanding the Makefile) Please study the `Makefile` used in the previous question. Here, we indicate the Terminal prompt by "*nn* >" where *nn* are successive counter values. Explain in your own words what each of the following commands is doing:

```
1 > make h
2 > make
3 > make p=Hello r
4 > make p=Hello r0
5 > make p=Hello r1
```

PART B: Birthday Paradox

Write your solution to PART B in a file called `BdayParadox.java` to be included in your Zip file submission.

B.1 The **Birthday Paradox** says that in any group of 23 or more people, the chance of two people sharing a birthday is more than 50%. In fact, the probability is 50.7%. Since our class has over 70 students, you can be sure that many of you share a birthday with one or more classmates! We will test this idea empirically.

You must write a class `BdayParadox`. The standard command line arguments are `ss` (seed), `nn` (size), and `mm` (mode). When the user does not provide any values for one or more of these arguments, your Java program must provide these values:

`ss=111`, `nn=23`, `mm=0`.

First we explain the concept of a **trial** (also known as an **experiment**): a trial consists in randomly generating `nn` many random birthdays (a "birthday" may be viewed as an integer in the range $[1, 365]$, i.e., a day in a non-leap year). If all the `nn` birthdays are distinct, the trial is a **failure** (the Birthday Paradox seems wrong!). Otherwise there exists some birthday is duplicated in the trial, so the trial is a **success**.

B.2 The behavior of your program depends on the mode (`mm`). We are interested in two modes: `mm=0` and `mm=1`. First, let us explain mode 0 (`mm=0`):

- Use `ss` as seed for a random number generator (called `rgen`). As usual, if `ss=0`, you must NOT provide any seed for `rgen` (then it is truly "random").
- Using `rgen`, you first generate an array of `nn` integers (representing birthdays). This represents a trial.
- Print the list of `nn` birthdays in the trial. IMPORTANT: *each birthday MUST be printed in the "month day" format.* For instance, day 1 is printed as "Jan 1", and day 365 is printed as "Dec 31". Also day 60 should print as "Mar 1" (since January has 31 days and February has 28 days). Print 10 days per row.
- Finally, check if the trial is a success or failure. If success, print a message identifying the first birthday that was duplicated; otherwise the message says "there are no duplicates".

We have created two targets called `test1` and `test2` in our Makefile that runs `BdayParadox` in mode 0. Here are their outputs:

> make test1

```
test 1:
/JavaHome/java.exe -classpath bin BdayParadox 111 23 0

===============The 23 birthdays are:
Jul23, Jan31, Mar14, Mar29, Aug28, Dec2, May20, Jul17, Oct25, Jan11,
Nov17, Dec6, Oct12, Nov16, Nov20, Aug27, Sep18, Nov13, Feb20, Aug4,
Apr1, Jul30, Nov16,
The 14-th bday and 23-th bday are equal: Nov16!
```

> make test2

```
test 2:
/JavaHome/java.exe -classpath bin BdayParadox 1111 23 0

===============The 23 birthdays are:
Dec23, Sep4, Feb15, Dec21, Aug9, Sep30, Oct13, Sep27, Jul8, Aug10,
Jan8, Dec16, Oct6, Sep20, Mar13, Apr15, Dec22, Mar14, Jun27, Apr24,
Dec3, Jul13, Dec10,
No duplicates found in 23 birthdays!
```

Your program must duplicate the outputs on these two tests *as closely as possible* (I mean literally!).

B.3 Now we will explain mode 1 (`mm=1`):

- Again `ss` serves as the seed for the random number generator `rgen`.
- We begin by generating a trial. As long as the trial is a failure, we make another trial. We stop as soon as we obtain a successful trial.
- Print a statement "the number of failed trials is `NNN`" where `NNN` is a non-negative integer.

Again, we have created two targets called `test3` and `test3` in our Makefile to run `BdayParadox` in mode 1. Here are their outputs:

> make test3

```
test 3:
/JavaHome/java.exe -classpath bin BdayParadox 111 20 1

===============
The total number of failed trials is 4
```

> make test4

```
test 4:
/JavaHome/java.exe -classpath bin BdayParadox 11 20 1

===============
The total number of failed trials is 2
```

Again, you must duplicate the outputs of tests 3 and 4 as closely as possible.