COS 126 **General Computer Science** **Fall 2021**

Programming Exam 1

Download the project zip now. Download the project zip which contains the files you will need for this exam. Open it in `IntelliJ`, the same way as you do for your assignments. *Do not examine any files yet.*

Submission & Grading. As with the COS 126 assignments, you can and should submit your code multiple times, but only the last version will be graded.Your program will be graded on correctness, clarity (including comments), design, and reasonable efficiency. You will lose a substantial number of points if your program does not compile. The total number of points possible on this exam is 35.

Before the exam. Read this page of instructions before the exam begins, but do not start (by reading the next page) until instructed. You have fifty (50) minutes.

Resources. You may only use: your textbook, the booksite, your notes, your code from programming assignments, the code on the COS 126 course website, materials from lectures, labs and precepts, and existing Ed posts.

No communication is permitted (e.g., talking, texting, Ed, etc.) during the exam. We will be sitting outside the exam room if you have a question.

No electronic devices permitted (e.g., phones, headphones, calculators, etc.) during the exam, except with course staff. You may use your laptops and scratch paper only.

Do not discuss later. Due to multiple exam times, and various conflicts, some of your peers will take the exam at a different time. Do not discuss the exam contents with anyone (not even other students that you know already took the exam!) until after the graded exams are returned.

Do not remove this exam from the exam room. Return it to course staff at the end of the exam.

Print your name and NetID and Exam Room on this page, now:

Name: .... NetID: .... Exam Room: ....

Honor Code pledge. You must **electronically sign the honor code** by retyping the pledge below and then your name in the file `honor-code.txt`, and upload it to TigerFile as part of your solution. Please do this now.

"I pledge my honor that I have not violated the Honor Code during this examination."

Do not examine or start the exam until instructed to do so.

You will be implementing part of a game called *The Picnic Game.* In the game, one person thinks of a secret rule about what kinds of items can be brought to the picnic. After hearing about some items that can be brought (or not) to the picnic, players try to guess the rule. Today, we'll omit the guessing part and focus on the rules aspect. In this exam, you will implement up to 7 functions and test them individually:

1. `itemMessage` (7 pts): Returns a sentence stating whether an item can be brought to the picnic.
2. `rule1` (5 pts): Return `true` for words shorter than 6 characters (e.g. cat is allowed, but not banana).
3. `rule2` (9 pts): Return `true` for words containing the letter `'e'` (e.g. bubble, but not cat).
4. `isVowel` (6 pts): Return `true` if a character is a vowel (helper function for `rule3` and `rule4`).
5. `rule3` (5 pts): Return `true` for words ending in a vowel (e.g. bubble, but not cat).
6. `rule4` (2 pts*): Return `true` for words with more consonants than vowels (e.g. cat, but not banana).
7. `rule5` (1 pt*): Return `true` for words following the adage "I before E, except after C" (e.g. friend, but not glacier).

Each function stands alone, and the functions progressively increase in difficulty. Note (*): `rule4` and `rule5` are challenge functions and should only be tried after the other 5 functions have been implemented and tested.

We give you starter code in `Picnic.java` that compiles, runs, and includes all function declarations. You should modify the body of each function according to this specification. Note: *DO NOT* modify the signature/interface for *any* of the supplied functions – only the function body.

Part I (7 pts): `itemMessage`. In `Picnic.java`, we correctly implement a `rule0`, which returns `true` for words that start with the letter `'b'` (e.g. banana is allowed, but not collie). On the right is shown an example run of the program. The command line argument specifies which rule to use, in this case 0 (starts with `'b'`). The input to the program (item list) is read from the file `rule0.txt`.

```
% java-introcs Picnic 0 < rule0.txt
banana: true
collie: false
glacier: false
cat: false
bubble: true
```

Function `itemMessage` should create a message about an item, saying whether or not it is allowed to the picnic. As shown above, the provided code only prints the item and a boolean. Modify the body of the function so that it produces (exactly!) the more informative output shown on the right.

```
% java-introcs Picnic 0 < rule0.txt
Yes, you can bring a banana to the picnic.
No, you cannot bring a collie to the picnic.
No, you cannot bring a glacier to the picnic.
No, you cannot bring a cat to the picnic.
Yes, you can bring a bubble to the picnic.
```

Part II (25 pts): Rules 1-3 and `isVowel`. In `Picnic.java`, a separate function is declared for each rule (`rule1`, `rule2`...) but they all just return `false`. Each rule function has an argument – an array of characters for that word. For example "cat" is passed as the array {`'c'`,`'a'`,`'t'`}.

Rule 1 (5 pts): Words shorter than 6 chars. Modify `rule1` to return `true` for words that are shorter than 6 characters, and false for other words. When you compile and run the program it should produce the output shown on the right.

```
% java-introcs Picnic 1 < rule1.txt
Yes, you can bring a cat to the picnic.
Yes, you can bring a canoe to the picnic.
No, you cannot bring a banana to the picnic.
No, you cannot bring a glacier to the picnic.
```

Rule 2 (9 pts). Modify `rule2` so it returns `true` for **words containing the letter `'e'`** and `false` for all other words, as shown on the right.

```
% java-introcs Picnic 2 < rule2.txt
No, you cannot bring a banana to the picnic.
Yes, you can bring a collie to the picnic.
Yes, you can bring a glacier to the picnic.
```

Rule 3: Words ending in a vowel. To implement this rule, we first need a function to determine whether a character is a vowel – any of the letters `'a'`, `'e'`, `'i'`, `'o'`, or `'u'`. (Inputs are *lower case letters* only!)

As provided, function `isVowel` (6 pts) returns `false`. Modify it so it returns `true` for all characters that are vowels and `false` for all other characters.

Next, implement `rule3` (5 pts). *Hint: it should call* `isVowel`. After compiling and running, you should see the output shown on the right.

```
% java-introcs Picnic 3 < rule3.txt
Yes, you can bring a banana to the picnic.
Yes, you can bring a collie to the picnic.
No, you cannot bring a glacier to the picnic.
No, you cannot bring a cat to the picnic.
No, you cannot bring a neighbor to the picnic.
Yes, you can bring a bubble to the picnic.
```

Part III (3 pts): Challenge Rules. Congratulations if you made it this far! The two final rules may be difficult to code during a timed exam. They are meant to provide extra challenges only for those of you who have sufficient time. You will receive credit for these components only if your previous steps are all solved correctly. Moreover, even perfect solutions for these steps will be worth only a small fraction of the overall exam grade. So only attempt this part if you are confident in your previous solutions.

Rule 4: More consonants than vowels (2 pts). Words with more consonants than vowels are allowed to the picnic, but words with equal or fewer are not allowed. Consonants are letters that are not vowels, and as noted above you may assume all inputs are lower case characters only. *Hint:* call `isVowel`.

```
% java-introcs Picnic 4 < rule4.txt
No, you cannot bring a banana to the picnic.
No, you cannot bring a collie to the picnic.
Yes, you can bring a glacier to the picnic.
Yes, you can bring a cat to the picnic.
Yes, you can bring a receipt to the picnic.
```

Rule 5: The adage "I before E, except immediately after C" (1 pt). (*Warning:* this rule is difficult!) The adage helps people remember how to spell words in English, and it works in most cases (e.g. friend). However, there are some exceptions where it fails (e.g. glacier). Only words that follow the adage are allowed to the picnic (e.g. friend). Words where the adage fails are not allowed (e.g. glacier). Any word that contains consecutive letters `'i'` and `'e'` should have them in that order, *unless* they appear immediately after `'c'` – in which case they should be in the order `'c'`, `'e'`, and then `'i'`. If the letters are in the correct order according to this adage, the word is allowed; if they do not appear in the correct order, the word is not allowed. Any word *without* I-E pairs is allowed. See the example cases shown below. Be careful about some tricky cases: (1) the I-E pair might be at the start of the word (e.g. "eightball"); and (2) it's possible for a word to have more than one I-E pair (e.g. "weirdie").

```
.                                         % java-introcs Picnic 5 < rule5.txt
This word has no I-E pair.................. Yes, you can bring a banana to the picnic.
This word has I then E (no C).............. Yes, you can bring a friend to the picnic.
This word has E then I (no C).............. No, you cannot bring a neighbor to the picnic.
This word has I then E, after C............ No, you cannot bring a glacier to the picnic.
This word has E then I, after C............ Yes, you can bring a receipt to the picnic.
This word has E then I at start............ No, you cannot bring a eightball to the picnic.
This word has BOTH E then I AND I then E... No, you cannot bring a weirdie to the picnic.
```