1 Introduction

Item-0. NJIT-ID (last 4 digits) and myUCID. Retrieve the last four digits of your NJIT-ID (NOT YOUR SSN): Document 0, Section 5, subsection 2 talks about this. If you don't know your NJIT-ID, login to `my.njit.edu` to retrieve it. For the sake of an example, a homework will be denoted by `hw`, and the last four digits of an ID as `1234`. In the remainder, we will be using the underscore symbol rather than the dash (minus) symbol, where needed. **The symbol _ is the underscore symbol, not the minus/dash - symbol.** (Java detests dashes and prefers underscores in identifiers, so we use the latter for consistency.) The Programming Project (PrP) is denoted as `prp`, or `pp1` and `pp2` to distinguish option 1 from option 2.

2 Canvas: Rules and filenames or filetypes.

RULE-0. Accidental submit and resubmit. The last of multiple (hw or PrP) submissions will be graded. The limit however would be kept low (possibly three). Do not send files by email, they will be discarded.

RULE-1. Canvas and Homeworks. You are expected to submit a homework through Canvas. No file submission. Just use a textbox to write your answer(s) or respond as requested (eg multiple choice).

RULE-2. Canvas and PrP. You are expected to upload and submit the PrP through Canvas. Only a single .tar or .zip file. Canvas will reject submissions greater than (or equal to about) 5MiB or 5MB.

RULE-3: File-name of a PrP submission. The only permissible archive formats are `zip` or `tar` for the PrP; one file per deadline available. The filename suffix would be .zip or .tar as applicable. The first three characters is pp1 for option 1 and pp2 for option 2, followed by say a _1234 as a filetype suffix (Item-0 talks about 1234s). If there is only one deadline for both option 1 and 2, use prp instead of pp1, pp2.

Example. If your NJIT-ID ends with 1234, a prp filename could be `pp1_1234.tar` or `pp2_1234.zip`. If there is only one deadline for option 1 and 2, use eg. `prp_1234.tar` then.

The zip/tar archive contains ALL the source of ALL options submitted and the single text file allowed. When files are dearchived, make sure no directories (folders) are created (MAC-OSX creates such) nor are there hidden files or executable or binary files of any type. In linux you may check for hidden directories with `ls -al`. Abide to the naming conventions of Section 3. For code that generates output files (eg a xyz.java generates binary file xyz.class) read carefully all of this handout to observe naming conventions.

A single text file named for example `pp1_1234.txt` within the archive contains either a
”HANDOUT 2 and PrP ADHERED; NO BUGS TO REPORT”
in capital case as shown (no quotation marks needed), or it contains compilation and execution instructions, a bug report and other useful information. The first line of this file must be as in Section 3 below.

Before you submit your archive test it on an OSL/AFS machine. Test it means: dearchive, compile, run on that machine. Compiling on a GUI environment is different from using the grading environment. For example manipulating text files on Windows or MAC-OSX can cause newline, linefeed problems.

RULE-4: Submissions that deviate from RULE-1 through RULE-3 get 0 points.

3 File and Class names; File Identification

Include your name and last four digits of your NJIT-ID in every file you submit. Every filename or (source code) class name must end with the last four digits of your ID. So use `Myclass1234` or `Myclass_1234` instead of `Myclass` stored in `Myclass1234.java` or `Myclass_1234.java` respectively.

4 Testing and Grading (of the PrP)

4.0 OSL or AFS machine access OFF-CAMPUS. In order to access an OSL machine (eg osl22.njit.edu) or AFS machine (eg afsconnect2.njit.edu) from OFF-CAMPUS, download a VPN client (NJIT supplies one through ist.njit.edu) and install and then activate it as needed by connecting to NJIT. Then use a secure shell client. The site ist.njit.edu contains relevant info or search through Google for example ('NJIT ssh', or 'NJIT VPN'). OSL or AFS machines are Linux machines. They are accessible through the command-line. Dearchiving and compilation and execution are through the command line (eg unzip, tar, gcc, g++, javac and java).

4.1 Read requirements carefully. Both options require **command line processing** and **file-based I/O**. If you are not familiar with them figure this out early in the semester and preferably by the midterm. Submissions that cannot handle command line processing or file-based input/output correctly risk of getting 0 points as ordinary testing will not work.

4.2 Testing and Debugging. Make sure your code dearchives (no sub/directories/folders created, hidden or not), compiles and runs on an AFS machine such as `afsconnect1.njit.edu`, `afsconnect2.njit.edu` or `osl22.njit.edu`. This also means you should edit text files, if needed, on AFS using AFS editing tools; avoid remote editing. Do a `gcc -v` or `g++ -v` or `javac -version` or `java -version` to confirm and report in the .txt file the version of compiler used. Versions available on afs are found with a `module avail` and loaded with say a `module load gcc/4.9.2`. If compilation is to proceed in a certain file sequence add a note in the `pp1_1234.txt` or `pp2_1234.txt` file, as needed. DO NOT DO testing 'remotely' using 'software of the remote platform' to edit files on 'AFS'. If you do not know what a newline becomes in Linux/Unix, Windows, MAC-OSX, be prepared for nasty surprises.

4.3 File types to submit. C or C++ files with `.c` or `.h` or `.cc` or `.cpp` or Java files with `.java` are acceptable. A single text file `pp1_1234.txt` or `pp2_1234.txt` must be included per RULE-3. The more info you have in it the less chances you have to get a 0.

4.4 Presence of directory structure is not allowed; file types as in 4.3. Inclusion of binary files (.jar, .class, etc) or other file types than those allowed in 4.3 above will **penalize you 80 points**. Beware of MAC OSX: it has a tendency of generating 0pt PrP submissions because it creates hidden directories. Do not skip step 4.2 above.

4.5 Grading For a HW, grading is more or less straighforward. For PrP, the grader will first decide and create testing instance(s) and grade your submission based on whether it passes successfully or not those testing instances using the specified interface (eg command-line processing). If your code does not pass any of these testing instances, it will get 0 points, unless there is a detailed bug report that YOU HAVE PROVIDED in `pp1_1234.txt` or `pp2_1234.txt`. Any information you provide there, it will help the grader to test your code on some reasonable inputs consistent with the default ones used for everybody else. If no information is provided by you, the grader will NOT read your code.

4.6 Grade and Canvas Grading. The PrP or HW grade will be made available in Canvas. Ignore canvas grade accumulations; canvas has no clue about the course grading scale or scheme. The only deadline is before (12-o'clock) noon of the day specified in the calendar of Document 1 (Syllabus).]

4.7 How to get a 0. If a submission is in .rar, or uses HashMaps or sorting of any type, or performs linear-time operations where a constant-time solution is possible (eg using Vector operations inefficiently) will get you 0 pts: **Do not complain then.** ■