AP® COMPUTER SCIENCE A

GENERAL SCORING GUIDELINES

Apply the question assessment rubric first, which always takes precedence. Penalty points can only be deducted in a part of the question that has earned credit via the question rubric. No part of a question (a, b , c) may have a negative point total. A given penalty can be assessed only once for a question, even if it occurs multiple times, or in multiple parts of that question. A maximum of 3 penalty points may be assessed per question.

1-Point Penalty

(w) Extraneous code that causes side effect (e.g., printing to output, incorrect precondition check)

(x) Local variables used but none declared

(y) Destruction of persistent data (e.g., changing value referenced by parameter)

Mr Lee's 1-Point Penalty:

- Inefficient, "long winded" or "messy" difficult to understand code which takes longer to write than standard more efficient solutions.
    - *In an exam you need to save time by writing quickly hand writable efficient code which is easy for AP readers to understand.*

No Penalty

- Extraneous code with no side effect (e.g., precondition check, no-op)
- Spelling/case discrepancies where there is no ambiguity*
- Local variable not declared provided other variables are declared in some part
- Keyword used as an identifier
- Common mathematical symbols used for operators (x • ÷ ≤ ≥< > ≠)
- `=` instead of `==` and vice versa
- Missing `{}` where indentation clearly conveys intent
- Missing `()` around `if` or `while` conditions

** Spelling and case discrepancies for identifiers fall under the "No Penalty" category only if the correction can be unambiguously inferred from context; for example, "total" instead of "totl". As a counterexample, that if the code declares "int G=99 , g=O; ", then uses "while (G < 10) " instead of "while (g < 10) ", the context does not allow for the reader to assume the use of the lower-case variable.*

Strings – Menu FRQ

SHOW ALL YOUR WORK. REMEMBER THAT PROGRAM SEGMENTS ARE TO BE WRITTEN IN JAVA.

Assume that the following variables have been properly declared and initialized.

- a `boolean` variable named `rsvp`
- an `int` variable named `selection`, where `1` represents `"beef"`, `2` represents "`chicken`", `3` represents "`pasta`", and all other values represent "`fish`"
- a `String` variable named `option2`

(a) Write a code segment that will store a dinner selection in a `String` variable named `option1` based on the values of `rsvp` and `selection`. The intended behavior of the code segment is described below.

If `rsvp` is `true`, the code segment should store in `option1` a `String` indicating the person's attendance and food choice. For example, if `rsvp` is `true` and `selection` is `1`, the following `String` should be stored in `option1`.

```
"Thanks for attending. You will be served beef."
```

If `rsvp` is `false`, the following `String` should be stored in `option1`, regardless of the value of `selection`.

```
"Sorry you can't make it."
```

Write the code segment below. Your code segment should meet all specifications and conform to the examples.

(b) Write a code segment that will print `true` if the strings `option1` and `option2` contain the same values and will print `false` otherwise.