Question 1 of 3

Recall that, in lecture, we saw an implementation of a spell checker in Python.

a) Why were we able to implement a spell checker in Python using fewer lines of code than it took to implement a spell checker in C?
b) Why did the spell checker in Python likely run slower than the spell checker you wrote in C?

Answers

a) The Python programming language is a higher-level programming language than C, which gives us access to abstractions that weren't available in languages like C. As a result, certain operations that may involve many steps (like hashing a word and inserting it into a hash table) can be expressed in Python using just a single line.
b) Answers may vary. Possibilities include:
    i) Our C program was compiled to machine code, and those instructions were executed by the computer directly. In Python, our program is interpreted, which results in some additional overhead since the interpreter must first interpret the Python program before executing the instructions.
    ii) When running a Python program, Python does additional work behind-the-scenes, like managing memory and checking for exceptions (i.e. errors), which takes additional time.

Question 2 of 3

Recall from lecture that, whereas C programs are compiled, Python programs are interpreted. In your own words, what does it mean for a programming language to be interpreted language?

Answer

When a language like Python is interpreted, a program called an interpreter reads a source code file, parses the syntax of the file, and executes the instructions. By contrast, C programs are compiled, which means the source code file is first translated into machine code that is directly executed by the computer; this process does not happen with Python programs.

Question 3 of 3

Recall that, in C, to get a positive integer between 1 and 8, inclusive, we could use code like the below.

```
1  int n;
2  do
3  {
4      n = get_int("Height: ");
5  }
6  while (n < 1 || n > 8);
```

In Python, there are no `do while` loops, so we would express the same idea as the below.

```
1  while True:
2      n = get_int("Height: ")
3      if n >= 1 and n <= 8:
4          break
```

Explain how these blocks of code are logically equivalent, as by explaining how each works line by line.

Answer

In the C program, line 1 declares a new variable called `n`. Line 2 starts a `do while` loop that will continually prompt for a new integer `n` (line 4) so long as that integer `n` is less than `1` or greater than `8` (given by the condition on line 6). Thus, when the loop finally ends, that must be because the condition is false, meaning that `n` is neither less than `1` nor greater than `8`, which means the result must be between `1` and `8`, inclusive.

In the Python program, line 1 starts an infinite loop. In the loop, line 2 prompts the user to type in an integer `n`. If that integer is valid, as by being between `1` and `8`, inclusive, the loop is exited (by breaking out of the loop on line 4). Otherwise, if the input isn't between `1` and `8`, inclusive, the loop repeats again. This process will continue to repeat until the result is between `1` and `8`, inclusive.

Thus, in both cases, the result is that `n` stores a value between `1` and `8`, inclusive.