ENCM 339 Fall 2015 Midterm: Section 3, Part c Solutions

This solution follows the **imprecise** hint given on the question paper:

```
int same_letters(const char *s, const char *t)
{
  int result = 1;
  int si = 0, ti = 0;  // indexes into s and t

  while (1) {
    while (s[si] != '\0' && !isalpha(s[si]))
      si++;
    while (t[ti] != '\0' && !isalpha(t[ti]))
      ti++;

    // Change result to 0 if two letters don't match or we're
    // comparing a letter to '\0'.
    if (s[si] != t[ti]) {
      result = 0;
      break;
    }

    // result stays 1 if we're at the ends of both strings.
    if (s[si] == '\0')
      break;

    // If we're here, two letters just matched, so we need to step
    // forward in both strings.
    si++;
    ti++;
  }

  return result;
}
```

Note that if the statements `si++;` and `ti++;` do not appear at the end of the outer loop, the outer loop may be infinite.

Many students came up with something like this correct and good solution, which doesn't really use the hint:

```
int same_letters(const char *s, const char *t)
{
  int si = 0, ti = 0;

  // Outer loop traverses the s string.
  while (s[si] != '\0') {
    if (isalpha(s[si])) {

      // Inner loop finds letter or '\0' in the t string.
      while (t[ti] != '\0' && !isalpha(t[ti]))
        ti++;
      if (s[si] != t[ti])  // if mismatched letters or t[ti] == ’\0’
        return 0;

      ti++;
    }  // End of inner loop.
    si++;
  }    // End of outer loop.

  // To return 1, we need to have used up all the letters in the t string.
  while (t[ti] != ’\0’) {
    if (isalpha(t[ti]))
      return 0;
    ti++;
  }
  return 1;
}
```