AP Computer Science
Dr. Paul L. Bailey

Responses 03/19
Wednesday, March 19, 2020

Question 1. What do I input for the String path in the load and content methods?

Answer. You need to have a text file in a place that you know where it is in your computers file system. For example, I have a file named `gettysburg.txt` in the directory `C:\Text`. So, I used

```
public static void test()
{
    String path = "C:/Text/gettysburg.txt";
    String s1 = Text.loadText(path);
    String s2 = Cipher.strip(s1);
    System.out.println(s2);
    System.out.println();

    int key = 5;
    String s3 = Cipher.caesar(s2, key);
    System.out.println(s3);
    System.out.println();
}
```

□

Question 2. To scan for non-letters, would we find ascii values and test whether they are within a bound?

Answer. It is possible to compare characters, such as

```
if (c >= 'A' && c <= 'Z') <do something>
```

It is also possible to add and subtract characters (Java converts to the ASCII value):

```
int v = ((c - 'A' + k) % 26) + 'A';
```

□

Question 3. I don't know where to start with it.

Answer. Start by typing in the code to load and strip files.

If you get that to work, copy the `strip(String text)` method and rename it `caesar(String text, int key)`. Insert a line of code to modify one letter at a time. □

Question 4. Are we allowed to convert the plaintext string to an array?

Answer. Yes, if that helps you. □

Question 5. I am having a problem with the deaffine method. If I have a = 2 and b = 1, this means that 'A' will map to 'B', but so will 'N'. So the string "AN" will become "BB", and when put through deaffine it becomes "AA", so the 'N' is lost. How should I fix this?

Answer. You cannot use an even number for your a, since then the mapping is "not injective". See below. □

Question 6. For deaffine, are the inverses of integers a and b integers such that a a inverse (mod 26)=1?

Question 7. While we are decrypting the affine cipher, do we only put in odd values for the "a" value, as it is not possible to use even values when finding the modular inverse? Therefore, when we encrypting the affine cipher, should we only accept values in which "a" is odd?

Answer. Let $a, n \in \mathbb{Z}$ with $n \geq 2$. This means that a and n are integers.

We say that a is *invertible modulo* n if there exists $b \in \mathbb{Z}$ such that $ab = 1$ modulo n. We can pick a and b to be in the range 1 to $n - 1$.

It is a theorem that a is invertible modulo n if and only if a and n are relatively prime; that is, if $\gcd(a, n) = 1$.

Thus the integers which are invertible mod 26 are the odd number other than 13.

I used this bit of code to find the inverses:

```
public static void findInverses()
{
    for (int n = 1; n < 26; n += 2)
    {
        if (n == 13) continue;

        for (int v = 1; v < 26; v += 2)
        {
            if (v == 13) continue;

            if ((n * v) % 26 == 1)
            {
                System.out.printf("n: %d  v: %d\n", n, v);
            }
        }
    }
}
```

The affine cipher $x \mapsto ax + b$ requires that a be invertible modulo 26.

To find the decryption key for the affine cipher, write $y = ax + b$ and solve for y to get $x = a^{-1}(y - b) = a^{-1}y + a^{-1}(-b)$.

Thus inverse of the key (a, b) is $(a^{-1}, -a^{-1}b)$, where a^{-1} is the inverse of a modulo 26. □