Goals:

- Practice getting around the command line compiling and running Java programs.
- Practice getting around in and using GitHub.
- Explain some key concepts of the Advanced Encryption System (AES).
- To get you familiar with one of the best cryptosystems of our time.
- Work hard for lab points.

Description:

It is public domain information that the U.S. Government allows usage of the Advanced Encryption Standard (AES) to protect SECRET and TOP SECRET information depending on the key-length used. One important part of the AES is how it produces keys on every round of encryption. The instructions on how to generate the 11 round keys $K_i$, for $i = 0, 1, \ldots, 10$, from the original 128-bit key $K_e$ are the following:

1. Let $K_e \in \mathbb{N}_{\text{hex}}^{4 \times 4}$ be the 128-bit encryption key in the form of a $4 \times 4$ matrix as follows:

$$
K_e = \begin{bmatrix}
k_{0,0} & k_{0,1} & k_{0,2} & k_{0,3} \\
k_{1,0} & k_{1,1} & k_{1,2} & k_{1,3} \\
k_{2,0} & k_{2,1} & k_{2,2} & k_{2,3} \\
k_{3,0} & k_{3,1} & k_{3,2} & k_{3,3}
\end{bmatrix}
$$

where every entry $k_{i,j} \in \mathbb{N}_{\text{hex}}$ contains a pair of hexadecimal digits, e.g. $k_{1,2} = \text{E0}$. Similarly, let $W \in \mathbb{N}_{\text{hex}}^{4 \times 44}$ be a $4 \times 44$ matrix of the following form:

$$
W = \begin{bmatrix}
w_{0,0} & w_{0,1} & w_{0,2} & \ldots & w_{0,43} \\
w_{1,0} & w_{1,1} & w_{1,2} & \ldots & w_{1,43} \\
w_{2,0} & w_{2,1} & w_{2,2} & \ldots & w_{2,43} \\
w_{3,0} & w_{3,1} & w_{3,2} & \ldots & w_{3,43}
\end{bmatrix}
$$

where every entry $w_{i,j} \in \mathbb{N}_{\text{hex}}$ contains a pair of hexadecimal digits, e.g. $w_{1,2} = \text{9A}$. Then, let $k(i), w(j) \in \mathbb{N}_{\text{hex}}^4$ denote column vectors constructed from the $i$-th column of $K_e$ and $j$-th column of $W$, respectively as follows:

$$
k(i) = \begin{bmatrix}
k_{0,i} \\
k_{1,i} \\
k_{2,i} \\
k_{3,i}
\end{bmatrix}, \quad w(j) = \begin{bmatrix}
w_{0,j} \\
w_{1,j} \\
w_{2,j} \\
w_{3,j}
\end{bmatrix}.
$$
Table 1: S-box to transform bytes. For example, if the input byte is 8B the corresponding transformed output byte will be 3D, because you take 8 as the row and B as the column in the table, which points to 3D.

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>63</td>
<td>7C</td>
<td>77</td>
<td>7B</td>
<td>F2</td>
<td>6B</td>
<td>6F</td>
<td>C5</td>
<td>30</td>
<td>01</td>
<td>67</td>
<td>2B</td>
<td>FE</td>
<td>D7</td>
<td>AB</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>CA</td>
<td>82</td>
<td>C9</td>
<td>7D</td>
<td>FA</td>
<td>59</td>
<td>47</td>
<td>F0</td>
<td>AD</td>
<td>D4</td>
<td>A2</td>
<td>AF</td>
<td>9C</td>
<td>A4</td>
<td>72</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>B7</td>
<td>FD</td>
<td>93</td>
<td>26</td>
<td>36</td>
<td>3F</td>
<td>F7</td>
<td>CC</td>
<td>34</td>
<td>A5</td>
<td>E5</td>
<td>F1</td>
<td>71</td>
<td>D8</td>
<td>31</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>04</td>
<td>C7</td>
<td>23</td>
<td>C3</td>
<td>18</td>
<td>96</td>
<td>05</td>
<td>9A</td>
<td>07</td>
<td>12</td>
<td>80</td>
<td>E2</td>
<td>EB</td>
<td>27</td>
<td>B2</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>09</td>
<td>83</td>
<td>2C</td>
<td>1A</td>
<td>1B</td>
<td>6E</td>
<td>5A</td>
<td>A0</td>
<td>52</td>
<td>3B</td>
<td>D6</td>
<td>B3</td>
<td>29</td>
<td>E3</td>
<td>7F</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>53</td>
<td>D1</td>
<td>00</td>
<td>ED</td>
<td>20</td>
<td>FC</td>
<td>B1</td>
<td>5B</td>
<td>6A</td>
<td>CB</td>
<td>BE</td>
<td>39</td>
<td>4A</td>
<td>4C</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>D0</td>
<td>EF</td>
<td>AA</td>
<td>FB</td>
<td>43</td>
<td>4D</td>
<td>33</td>
<td>85</td>
<td>45</td>
<td>F9</td>
<td>02</td>
<td>7F</td>
<td>50</td>
<td>3C</td>
<td>9F</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>51</td>
<td>A3</td>
<td>40</td>
<td>8F</td>
<td>92</td>
<td>9D</td>
<td>38</td>
<td>F5</td>
<td>BC</td>
<td>B6</td>
<td>DA</td>
<td>21</td>
<td>10</td>
<td>FF</td>
<td>F3</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>CD</td>
<td>0C</td>
<td>13</td>
<td>EC</td>
<td>5F</td>
<td>97</td>
<td>44</td>
<td>17</td>
<td>C4</td>
<td>A7</td>
<td>7E</td>
<td>3D</td>
<td>64</td>
<td>5D</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>60</td>
<td>81</td>
<td>4F</td>
<td>DC</td>
<td>22</td>
<td>2A</td>
<td>90</td>
<td>88</td>
<td>46</td>
<td>EE</td>
<td>B8</td>
<td>14</td>
<td>DE</td>
<td>5E</td>
<td>0B</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>E0</td>
<td>32</td>
<td>3A</td>
<td>0A</td>
<td>49</td>
<td>06</td>
<td>24</td>
<td>5C</td>
<td>C2</td>
<td>D3</td>
<td>AC</td>
<td>62</td>
<td>91</td>
<td>95</td>
<td>E4</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>E7</td>
<td>C8</td>
<td>37</td>
<td>6D</td>
<td>8D</td>
<td>D5</td>
<td>4E</td>
<td>A9</td>
<td>6C</td>
<td>56</td>
<td>F4</td>
<td>EA</td>
<td>65</td>
<td>7A</td>
<td>AE</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>BA</td>
<td>78</td>
<td>25</td>
<td>2E</td>
<td>1C</td>
<td>A6</td>
<td>B4</td>
<td>C6</td>
<td>E8</td>
<td>DD</td>
<td>74</td>
<td>1F</td>
<td>4B</td>
<td>BD</td>
<td>8B</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>70</td>
<td>3E</td>
<td>B5</td>
<td>66</td>
<td>48</td>
<td>03</td>
<td>F6</td>
<td>0E</td>
<td>61</td>
<td>35</td>
<td>57</td>
<td>B9</td>
<td>86</td>
<td>C1</td>
<td>1D</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>E1</td>
<td>F8</td>
<td>98</td>
<td>11</td>
<td>69</td>
<td>D9</td>
<td>8E</td>
<td>94</td>
<td>9B</td>
<td>1E</td>
<td>87</td>
<td>E9</td>
<td>CE</td>
<td>55</td>
<td>28</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>8C</td>
<td>A1</td>
<td>89</td>
<td>0D</td>
<td>BF</td>
<td>E6</td>
<td>42</td>
<td>68</td>
<td>41</td>
<td>99</td>
<td>2D</td>
<td>0F</td>
<td>B0</td>
<td>54</td>
<td>BB</td>
<td></td>
</tr>
</tbody>
</table>

2. Now, we take the AES key and make it be the first four columns of $W$ by making $w(0) = k(0)$, $w(1) = k(1)$, $w(2) = k(2)$, and $w(3) = k(3)$. That creates the first four columns of $W$, and that constitutes round $i = 0$.

3. For the other 40 columns we do the following:

(a) If the column index $j$ is not a multiple of 4. We XOR the fourth past and last column with respect to $j$, as denoted in the following equation:

$$w(j) = w(j - 4) \oplus w(j - 1)$$

(b) If the column index $j$ is a multiple of 4, this indicates that we are starting a new round $i$, but we can always know in what round we are by computing $i = \left\lfloor \frac{j}{4} \right\rfloor$; and we proceed as follows:

i. For the construction of $w(j)$ we will use the elements of the previous column $w(j - 1) = [w_{0,j-1} \ w_{1,j-1} \ w_{2,j-1} \ w_{3,j-1}]^T = w_{\text{new}}$ and store them into a temporary vector $w_{\text{new}}$.

ii. Then we perform a shift to the left as follows: $w_{\text{new}} = [w_{1,j-1} \ w_{2,j-1} \ w_{3,j-1} \ w_{0,j-1}]^T$.

iii. Next we transform each of the four bytes in $w_{\text{new}}$ using an S-box function $S(\cdot)$ (supported by Table 1) as follows $w_{\text{new}} = \left[S(w_{1,j-1}) \ S(w_{2,j-1}) \ S(w_{3,j-1}) \ S(w_{0,j-1})\right]^T$.

iv. Get the Rcon($i$) constant for the $i$-th round by using the look-up Table 2.

v. Perform an XOR operation using the corresponding round constant obtained in the previous step as follows: $w_{\text{new}} = \left[(\text{Rcon}(i) \oplus S(w_{1,j-1})) \ S(w_{2,j-1}) \ S(w_{3,j-1}) \ S(w_{0,j-1})\right]^T$.

vi. Finally, $w(j)$ can be defined as follows:

$$w(j) = w(j - 4) \oplus w_{\text{new}}.$$  

4. Every round key is then composed of 4 successive readings of the columns of $W$. E.g., round zero is composed of $w(0)$, $w(1)$, $w(2)$, and $w(3)$; round one will be composed of $w(4)$, $w(5)$, $w(6)$, and $w(7)$; and so on.
Table 2: Data to retrieve the $i$-th round constant Rcon($i$). For example, Rcon(1) = 1, the Rcon(2) = 2, Rcon(3) = 4, and Rcon(9) = 1B.

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>8D</td>
<td>01</td>
<td>02</td>
<td>04</td>
<td>08</td>
<td>10</td>
<td>20</td>
<td>40</td>
<td>80</td>
<td>1B</td>
<td>36</td>
<td>6C</td>
<td>D8</td>
<td>AB</td>
<td>4D</td>
<td>9A</td>
</tr>
<tr>
<td>2F</td>
<td>5E</td>
<td>BC</td>
<td>63</td>
<td>C6</td>
<td>97</td>
<td>35</td>
<td>6A</td>
<td>D4</td>
<td>B3</td>
<td>7D</td>
<td>FA</td>
<td>EF</td>
<td>C5</td>
<td>91</td>
<td>39</td>
</tr>
<tr>
<td>72</td>
<td>E4</td>
<td>D3</td>
<td>BD</td>
<td>61</td>
<td>C2</td>
<td>9F</td>
<td>25</td>
<td>4A</td>
<td>94</td>
<td>33</td>
<td>66</td>
<td>CC</td>
<td>83</td>
<td>1D</td>
<td>3A</td>
</tr>
<tr>
<td>74</td>
<td>E8</td>
<td>CB</td>
<td>8D</td>
<td>01</td>
<td>02</td>
<td>04</td>
<td>08</td>
<td>10</td>
<td>20</td>
<td>40</td>
<td>80</td>
<td>1B</td>
<td>36</td>
<td>6C</td>
<td>D8</td>
</tr>
<tr>
<td>AB</td>
<td>4D</td>
<td>9A</td>
<td>2F</td>
<td>5E</td>
<td>BC</td>
<td>63</td>
<td>C6</td>
<td>97</td>
<td>35</td>
<td>6A</td>
<td>D4</td>
<td>B3</td>
<td>7D</td>
<td>FA</td>
<td>EF</td>
</tr>
<tr>
<td>C5</td>
<td>91</td>
<td>39</td>
<td>72</td>
<td>E4</td>
<td>D3</td>
<td>BD</td>
<td>61</td>
<td>C2</td>
<td>9F</td>
<td>25</td>
<td>4A</td>
<td>94</td>
<td>33</td>
<td>66</td>
<td>CC</td>
</tr>
<tr>
<td>83</td>
<td>1D</td>
<td>3A</td>
<td>74</td>
<td>E8</td>
<td>CB</td>
<td>8D</td>
<td>01</td>
<td>02</td>
<td>04</td>
<td>08</td>
<td>10</td>
<td>20</td>
<td>40</td>
<td>80</td>
<td>1B</td>
</tr>
<tr>
<td>36</td>
<td>6C</td>
<td>D8</td>
<td>AB</td>
<td>4D</td>
<td>9A</td>
<td>2F</td>
<td>5E</td>
<td>BC</td>
<td>63</td>
<td>C6</td>
<td>97</td>
<td>35</td>
<td>6A</td>
<td>D4</td>
<td>B3</td>
</tr>
<tr>
<td>7D</td>
<td>FA</td>
<td>EF</td>
<td>C5</td>
<td>91</td>
<td>39</td>
<td>72</td>
<td>E4</td>
<td>D3</td>
<td>BD</td>
<td>61</td>
<td>C2</td>
<td>9F</td>
<td>25</td>
<td>4A</td>
<td>94</td>
</tr>
<tr>
<td>33</td>
<td>66</td>
<td>CC</td>
<td>83</td>
<td>1D</td>
<td>3A</td>
<td>74</td>
<td>E8</td>
<td>CB</td>
<td>8D</td>
<td>01</td>
<td>02</td>
<td>04</td>
<td>08</td>
<td>10</td>
<td>20</td>
</tr>
<tr>
<td>40</td>
<td>80</td>
<td>1B</td>
<td>36</td>
<td>6C</td>
<td>D8</td>
<td>AB</td>
<td>4D</td>
<td>9A</td>
<td>2F</td>
<td>5E</td>
<td>BC</td>
<td>63</td>
<td>C6</td>
<td>97</td>
<td>35</td>
</tr>
<tr>
<td>6A</td>
<td>D4</td>
<td>B3</td>
<td>7D</td>
<td>FA</td>
<td>EF</td>
<td>C5</td>
<td>91</td>
<td>39</td>
<td>72</td>
<td>E4</td>
<td>D3</td>
<td>BD</td>
<td>61</td>
<td>C2</td>
<td>9F</td>
</tr>
<tr>
<td>25</td>
<td>4A</td>
<td>94</td>
<td>33</td>
<td>66</td>
<td>CC</td>
<td>83</td>
<td>1D</td>
<td>3A</td>
<td>74</td>
<td>E8</td>
<td>CB</td>
<td>8D</td>
<td>01</td>
<td>02</td>
<td>04</td>
</tr>
<tr>
<td>08</td>
<td>10</td>
<td>20</td>
<td>40</td>
<td>80</td>
<td>1B</td>
<td>36</td>
<td>6C</td>
<td>D8</td>
<td>AB</td>
<td>4D</td>
<td>9A</td>
<td>2F</td>
<td>5E</td>
<td>BC</td>
<td>63</td>
</tr>
<tr>
<td>C6</td>
<td>97</td>
<td>35</td>
<td>6A</td>
<td>D4</td>
<td>B3</td>
<td>7D</td>
<td>FA</td>
<td>EF</td>
<td>C5</td>
<td>91</td>
<td>39</td>
<td>72</td>
<td>E4</td>
<td>D3</td>
<td>BD</td>
</tr>
<tr>
<td>61</td>
<td>C2</td>
<td>9F</td>
<td>25</td>
<td>4A</td>
<td>94</td>
<td>33</td>
<td>66</td>
<td>CC</td>
<td>83</td>
<td>1D</td>
<td>3A</td>
<td>74</td>
<td>E8</td>
<td>CB</td>
<td>8D</td>
</tr>
</tbody>
</table>

Your mission is to write a Java program in two files, DriverAES.java and AESCipher.java. The first file, AESCipher.java, will have a class for the AES cipher that in this case implements a method with the following signature `String[] roundKeysHex = aesRoundKeys(String KeyHex)` that will produce 11 round keys (one in every element of the string array) as explained in the above Description section. The input, KeyHex, is a length 16-hex string representation of the system key $K_e$. The output, roundKeysHex, will be an 11-row string array representation of all the round keys. Each element of roundKeysHex will contain a 16-hex string corresponding to each round key. You will also create two more methods to help you in your computations, one for reading the S-box with a signature `outHex = aesSBox(inHex)`, and another method to get each round’s constant with the following signature `outHex = aesRcon(round)`. So, your AESCipher.java file will have the implementation of all your AES code.

However, your DriverAES.java program will only test your implementation by calling `aesRoundKeys()` providing valid data.

The program should produce only keys as shown in the example below. Above all things try to make your code as efficient as possible. Look at the resources section below to copy-paste tables. Whenever possible try to work directly with hexadecimal values in Java.

**Input:**

Your driver should read the system key, $K_e$, from standard input, i.e., System.in. The key should be all in upper case
Output:

The output must be the 11 round keys, one in each row, all in upper case.

Sample Input 1:

```
546861747320BD79204B756E67204675
```

Sample Output 1:

```
546861747320BD79204B756E67204675
E232FCF191129188B159E4E6D679A293
56082007C71AB18F7643569A03AF7FA
D2600DE7157ABC686339E901C3031EFB
A11202C9B468BFA1D75157A01452495B
B1293B3305418592D210D232C6429B69
BD3DC87C47156A6C9527AC2E0E4E
CC96ED1674EAA031E863F24B2A8316A
8E51EF21FABBB4522E43DA0656954B6C
BFE2BF904559FAB2A16480B4F7F1CBD8
28FDDEF86DA4244ACC0A4FE3B316F26
```

Resources:

- Your textbook (Stanoyevitch)!
- Project submission guidelines for this course: [www.reev.us/mscs630s17/project_submission.html](http://www.reev.us/mscs630s17/project_submission.html)
- Coding style guidelines for this course: [www.reev.us/mscs630s17/style.html](http://www.reev.us/mscs630s17/style.html)
- “How to” use the command line “shell”: [www.reev.us/mscs630s17/shell.html](http://www.reev.us/mscs630s17/shell.html)
- Stack Overflow Java Tag: [http://stackoverflow.com/questions/tagged/java](http://stackoverflow.com/questions/tagged/java)

Submission:

- Push your work to your GitHub repository before the due date (see the top of this document). Remember to include your name, the date, and the assignment in the (copious, meaningful, and accurate) commit messages. Then double check your files are on GitHub.
- Make sure that your program (*.java) is in a folder called `labs/4/` inside your repository folder.