CMPE-013/L

Introduction to “C” Programming

Maxwell James Dunne
Battle Boats

Late days

\texttt{max(portner1, portner2)}

Lab 10
NMEA Protocol

<table>
<thead>
<tr>
<th>Character</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>'$'</td>
<td>The start-of-message identifier, always a dollar-sign</td>
</tr>
<tr>
<td>MESSAGE_ID</td>
<td>A 3-character string identifying the type of message.</td>
</tr>
<tr>
<td>','</td>
<td>A comma separates the MESSAGE_ID from the subsequent data</td>
</tr>
<tr>
<td>DATA1,DATA2,DATA3,...</td>
<td>A comma-separated list of data, all encoded as ASCII characters</td>
</tr>
<tr>
<td>'*XX'</td>
<td>A message ends with an asterisk and then a checksum byte encoded as two separate ASCII hexadecimal characters (like '0A'). This checksum is calculated from ALL bytes between the '$' and the '*'</td>
</tr>
<tr>
<td>'\n'</td>
<td>A newline character actually ends the string</td>
</tr>
</tbody>
</table>
checksum

$ \star \uparrow$
• Agent A generates a random 16-bit number that is its "guess" along with another 16-bit number that is used as the encryption key.

• Agent A then transmits a checksum of both its guess and key (which is an 8-bit XOR of all of their bytes) along with an encrypted version of its guess (which is a 16-bit XOR of the guess with the encryptionKey).

• During this time Agent B is doing the same thing.
• Once Agent A has received Agent B's encrypted guess and checksum, it transmits the unencrypted guess and the encryption key (and Agent B does the same).

• 5. Agent B can now verify Agent A's information by verifying both the checksum and the encryption key (and Agent A does the same).

• 6. Now both can agree on who should go first by having either guessed higher or lower than the other agent depending on if the XOR of the LSB of their guesses is 1 or 0.
Sample Guess

A

\[
\begin{align*}
\text{guess} & \times x \times x \times x \times x \times x \times x \\
\text{checksum} & = 8 \\
\text{guess} \oplus \text{key} & = \text{guess} \times x \times x \times x \\
\text{checksum} & = 16
\end{align*}
\]

B

\[
\begin{align*}
\text{guess} \oplus \text{key} & = \text{guess} \\
\text{checksum} & = 16
\end{align*}
\]
WAITING

Input != '$'
Return WAITING

Input == '$'
Index = 0
Return GOOD

RECORDING

Invalid hex character received
Return FAILURE

Input != '*'

Input == '*'
Sentence[Index] = Input
Index += 1
Return GOOD

FIRST_CHECKSUM_HALF

Valid hex character received
Save hex value as top 4-bits of stateData->checksum
Return GOOD

Invalid hex character received || checksum matching failed
Return FAILURE

SECOND_CHECKSUM_HALF

Input != '\n' ||
MSGID invalid
Return FAILURE

Input is valid hex char && checksum matching succeeded
Sentence[Index] = '"0'
Return GOOD

NEWLINE

Input is '\n' && MSGID is one of ('DET','CHA','COO','HIT')
Return one of PARSED
Any state

- Message parsing failed
  - Clear the OLEDS
  - Set OLED to ERROR_STRING_PARSING

- Negotiation data is invalid
  - Clear the OLEDS
  - Set OLED to ERROR_STRING_NEG_DATA

- Turn ordering was a tie
  - Clear the OLEDS
  - Set OLED to ERROR_STRING_ORDERING

- Won turn ordering
  - Set turn to MINE
  - Update screen

- Lost turn ordering
  - Set turn to THEIRS
  - Update screen

- Rec'd COO message && friendly boats left
  - Set turn to MINE
  - Register hit
  - Update screen
  - Send HIT message

- Rec'd HIT message && enemy boats left
  - Update knowledge of enemy field
  - Set turn to theirs

- Rec'd HIT message && no friendly boats left
  - Set turn to NONE
  - Update screen
  - Send HIT message

- Generated valid coordinates

- Rec'd COO message && no enemy boats left after hit
  - Send COO message

- Rec'd HIT message && No enemy boats left after hit
  - Set turn to NONE
  - Update screen

- Won turn ordering

- Wait_for_Guess

- Send_Guess

- Lost

- Won
```c
for(i in String)
    {ProtDecode(i)}
```

| Negotiation Data Set 1 | $CHA, 37348, 117*46
<table>
<thead>
<tr>
<th></th>
<th>$DET, 9578, 46222*66</th>
</tr>
</thead>
</table>
| Negotiation Data Set 2 | $CHA, 54104, 139*45
|                        | $DET, 32990, 21382*5e |
| Negotiation Data Set 3 | $CHA, 62132, 70*79
|                        | $DET, 52343, 16067*50 |
| Negotiation Data Set 4 | $CHA, 36027, 55*7a
|                        | $DET, 7321, 36898*6e  |
| HIT messages           | $HIT, 3, 8, 1*43
|                        | $HIT, 0, 2, 0*4b
|                        | $HIT, 2, 3, 1*49
|                        | $HIT, 5, 6, 4*4e
|                        | $HIT, 0, 3, 0*4a
|                        | $HIT, 1, 7, 1*4e
|                        | $HIT, 4, 8, 0*45
|                        | $HIT, 5, 3, 3*4c
|                        | $HIT, 0, 5, 0*4c
|                        | $HIT, 5, 6, 1*4b
|                        | $HIT, 1, 1, 1*48
|                        | $HIT, 1, 0, 0*48
|                        | $HIT, 5, 2, 5*4b
|                        | $HIT, 2, 8, 0*43
|                        | $HIT, 0, 6, 0*4f
|                        | $HIT, 5, 9, 0*45
|                        | $HIT, 2, 8, 2*41 |
Partners
someone else's
T W uno 32
Protocol or field

\[ \frac{3}{5} \]

\[ 2 \]