Mobeewave job interview

Develop in android studio. No need for fancy UI.
(https://developer.android.com/studio/install.html)

Java is not exactly an embedded language, but a variant of it used in the payment industry, Javacard. No need to do research about javacard, but use java as if you were in an embedded context. Moreover, in the embedded context at Mobeewave, knowledge of android studio is a must.

Refrain from using too much libraries in the 3 classes asked, remember we want to evaluate your coding capabilities in embedded context. Also, in javacard the biggest primitive available is the short (2 bytes), therefor don't use int, long etc... (arrays are available)

1) Please detail what java mechanisms you choose to use, to avoid, and why certain java mechanism are not relevant in an embedded context.
Mechanisms encompass object oriented paradigm, JVM own mechanisms (JIT, garbage collection, etc...)

2) The communication between the card and the terminal is done using small packet of data called APDU. The maximum size of an APDU is 250 Bytes. Data in APDU are usually encoded using the TLV format. And those data can be encrypted and authenticated.

- Create command APDU class:

https://www.wikiwand.com/en/Smart_card_application_protocol_data_unit

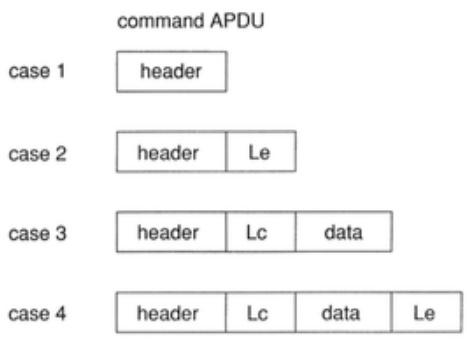
  - An APDU is byte array
  - Create multiple constructor (at least one per case) that returns the APDU as a byte array
  - Create a constructor that takes a byte array as an entry
  - Check validity. Given a byte array, perform a check that return which type of APDU it is.

Values in the header can be random, don't spend too much time Classes values, instructions values etc...

Valid APDU exemple: 80 E2 00 00 13 1002 10 0123456789ABCDEFFEDCBA9876543210
Invalid APDU exemple: 80 E2 00 00 10 1002 10 0123456789ABCDEFFEDCBA9876543210

Table 2.1 Command APDU structure

Mandatory header				**Optional body**		
CLA	INS	P1	P2	Lc	Data field	Le



- Create TLV class:
  https://www.wikiwand.com/en/X.690#/BER_encoding

You can define your type of tag, for sake of simplicity

- Build from value, from tlv...
- Find value
- Tag present

Parse this:
61 0A 4F 08 A0 00 00 01 51 00 00 00 61 0E 4F 0C A0 00 00 01 51 53 50 41 53 4B 4D 53 61 10 4F 0E A0 00 00 01 51 53 50 41 4C 43 43 4D 41 4D 61 10 4F 0E A0 00 00 01 51 53 50 41 4C 43 43 4D 44 4D 61 0F 4F 0D A0 00 00 01 51 53 50 41 53 33 53 53 44 61 0C 4F 0A A9 A8 A7 A6 A5 A4 A3 A2 A1 A0 61 0C 4F 0A A9 A8 A7 A6 A5 A4 A3 A2 A1 A1 61 0E 4F 0C A0 00 00 00 03 53 50 42 00 01 42 01 61 0E 4F 0C A0 00 00 01 51 53 50 43 41 53 44 00 61 0B 4F 09 A0 00 00 01 51 41 43 4C 00 61 12 4F 10 A0 00 00 00 77 01 07 82 1D 00 00 FE 00 00 02 00 61 12 4F 10 A0 00 00 02 20 53 45 43 53 45 53 50 52 4F 54 31 61 12 4F 10 A0 00 00 02 20 53 45 43 53 54 4F 52 41 47 45 31 61 12 4F 10 A0 00 00 02 20 15 03 01 03 00 00 00 41 52 41 43 61 0C 4F 0A A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 61 0C 4F 0A A0 A1 A2 A3 A4 A5 A6 A7 A8 AA 61 12 4F 10 A0 00 00 00 77 02 07 60 11 00 00 FE 00 00 FE 00 61 0B 4F 09 A0 00 00 01 51 43 52 53 00

You can check your result with this tool http://www.emvlab.org/tlvutils/

- Create an encryption class.

The chosen algorithm is a simple 2 bits shift to the right. The shift is done in a circular way, ie. The 2 rightmost bits that are lost due to the shifting are re-introduce to the left. Therefor there is no loss of information.

- Write an encryption and a decryption function.
- Encrypt / decrypt data in apdu, not the mandatory header

3) Using all the classes described above, display the result of parsing this byte stream:
80 E2 00 00 0A af 82 11 db db d9 08 12 9b d8

This is an APDU with an encrypted TLV in its data field.
a) get the data field from the apdu
b) decrypt data -> decrypted data is a TLV
c) parse TLV (1 byte long tag)

FYI: you will get an ASCII string after decryption and parsing.