Programming Task

Provide your most object oriented solution in Java. There is scope in the exercise to show inheritance, interfaces, abstraction, encapsulation and double dispatch.

You have to interface with a robot over Bluetooth. The command specification that the robot understands is below.

Write example Android code to make the robot move around a pentagon, taking a photograph at each corner. You should wait for the robot to complete each step in your program before you try to give it the next step.

The project includes a demonstration robot, Arthur, which you should connect to using the RobotConnection class. You need to pass the ApplicationContext to Arthur on instantiation.

```
Robot arthur = new Arthur(getApplicationContext());
RobotConnection conn = new RobotConnection(arthur)
```

Once configured, you can send and receive binary data from the robot using

```
conn.send(byte[] data);
byte[] data = conn.receive();
```

We've tested Arthur and think that he is compliant. Arthur is written in a deliberately liner, non-object oriented style, so do not copy Arthur code, design or patterns.

Arthur starts in the middle of a square arena, 50m x 50m. He is facing East.

Robot Command specification

Byte Index	Command	Length
1	Class Type	1
2	Instruction Type	1
3	Length of payload (int)	4
7	Payload	varies

There are 2 class types.
Movement has a value of 0x01 and **Utility** has a value of 0x02.

There are 5 instruction types
Move has a value of 0x01
Rotate has a value of 0x02
Wait has a value of 0x03
Take Photo has a value of 0x04
Speak has a value of 0x05

Move and **Rotate** instructions are in the **Movement** class; **Wait**, **Take Photo** and **Speak** are in

the **Utility** class.

Some Commands take parameters in the payload. These are specific to each command:

Move: The distance in meters, as a positive integer
Rotate: The number of degrees to rotate. Use a negative value to rotate anticlockwise, and a positive value to rotate clockwise. An integer value.
Speak: The text (ASCII) that the robot should say.
Wait: The duration in seconds that the robot will wait for. An integer value. Arthur will wait for a maximum of 5 seconds before getting bored and leaving, but the specification is for any positive integer value.

Envelopes

Commands must be sent in an Envelope. The envelope is defined as

Byte Index	Description	Length
1	Sequence Number (int)	4
5	Length of command (int)	4
9	Payload	varies

Each Envelope should have a higher sequence number than the previous one, preventing the robot responding to commands out of sequence.

As an example, an Envelope with the sequence number 9, containing a Command to rotate clockwise 90 degrees would look like this (hex data):

00 00 00 09 00 00 00 0A 01 02 00 00 00 04 00 00 00 5A

Responses

When the robot completes a task, it will respond with a Response message, in an Envelope. The sequence number of the Response should match the sequence number of the Envelope the Command was issued in.

Responses have the following format

Byte Index	Command	Length
1	Class Type	1
2	Instruction Type	1
3	Success (non-zero for successful)	1
4	Payload Length (optional)	4
8	Payload (optional)	Varies

If the response has no payload, then it does not need to include the payload length, and may be just 3 bytes long. The Response to **Take Photo** has the photograph data as the payload.