ISET SFAX AU 2018/2019 SEMESTRE I		
DEVOIR FINAL CORRECTION		
Classe : SEM31	*Matière : **D**éveloppement **M**obile **A**vancé*	*Nb pages : 8*
*Enseignant : **S**ouissi **H**afedh*		
Documents Non Autorisés	*Barème : 20 = 5 + 8 + 7*	*Durée : 1 heure 30 minutes*

Exercice1

"MainActivity"

```
private void analyser() {
        Bitmap bitmap = ((BitmapDrawable) imgVisages.getDrawable()).getBitmap();
        FaceDetector.Builder b = new FaceDetector.Builder(this);
        b.setTrackingEnabled(true);
        b.setLandmarkType(FaceDetector.ALL_LANDMARKS);
        b.setClassificationType(FaceDetector.ALL_CLASSIFICATIONS);
        FaceDetector detector = b.build();
        Frame frame = new Frame.Builder().setBitmap(bitmap).build();
        SparseArray<Face> faces = detector.detect(frame);
        int nbV=faces.size();
        int nbVO=0;
        int nbVR=0;
        for (int i = 0; i < faces.size(); ++i) {
            Face face = faces.valueAt(i);


            double xLC = 0, yLC = 0, xRC = 0, yRC = 0;
            for (Landmark l : face.getLandmarks()) {
                if (l.getType() == Landmark.LEFT_CHEEK) {
                    xLC = l.getPosition().x;
                    yLC = l.getPosition().y;
                }
                if (l.getType() == Landmark.RIGHT_CHEEK) {
                    xRC = l.getPosition().x;
                    yRC = l.getPosition().y;
                }
            }
            double hauteur=face.getHeight();
            double dJoues = Math.sqrt(Math.pow(xLC - xRC, 2) + Math.pow(yLC - yRC, 2));
            double cJH = dJoues / hauteur;
            Log.i("cJH","cJH:"+cJH);
            if(cJH>0.3f)
                nbVR++;
        }
        nbVO=nbV-nbVR;
        tvNbV.setText(nbV+"");
        tvNbVR.setText(nbVR+"");
        tvNbVO.setText(nbVO+"");
    }
```

Exercice2

"MainActivity"

```
private void ajouterEcouteur() {
        br= new BroadcastReceiver() {
            @Override
            public void onReceive(Context context, Intent intent) {
                actualiser(intent);
            }
        };
    }
    protected void actualiser(Intent intent) {
        double press=intent.getDoubleExtra("press", 0);
        String inter=intent.getStringExtra("inter");
        tvPress.setText(press+"");
        tvInter.setText(inter);

    }
    private void demarrerService() {
        Intent i = new Intent(this,PressService.class);
        startService(i);

    }
    private void arreterService() {
        Intent i = new Intent(this,PressService.class);
        stopService(i);
    }
    @Override
    protected void onResume() {
        demarrerService();
        registerReceiver(br, new IntentFilter(PressService.ACTION_PRESS));
        super.onResume();
    }
    @Override
    protected void onPause() {
        arreterService();
        unregisterReceiver(br);
        super.onPause();

    }
```

"PressService"

```
private void init() {
        smg = (SensorManager) getSystemService(SENSOR_SERVICE);
        press = smg.getDefaultSensor(Sensor.TYPE_PRESSURE);
        smg.registerListener(this, press, SensorManager.SENSOR_DELAY_UI);
    }
    @Override
    public void onDestroy() {
        smg.unregisterListener(this, press);
        super.onDestroy();
    }

    @Nullable
    @Override
    public IBinder onBind(Intent intent) {
        return null;
    }
```

```
    @Override
    public void onSensorChanged(SensorEvent event) {
        if (event.sensor.getType() == Sensor.TYPE_PRESSURE) {
            double press = event.values[0];
            String inter="";
            if(press<101)
                inter="Dépression";
            else
                inter="Anticyclone";

            Intent i = new Intent();
            i.setAction(ACTION_PRESS);
            i.putExtra("press", press);
            i.putExtra("inter", inter);

            sendBroadcast(i);
        }
    }
```

Exercice3

"ServDrone"

```
private void lancerServeur() {
    Runnable r = new Runnable() {
            @Override
            public void run() {
                demmarerServeur();
            }
        };
        Thread th = new Thread(r);
        th.start();
  }
  private void demarrerServeur() {
           try {
            ss = new ServerSocket(PORT);
            s = ss.accept();
            br = new BufferedReader(new InputStreamReader(s.getInputStream()));
            while (true) {
                String cmd;
                cmd = br.readLine();
                executerCommande(cmd);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
  }
  private void executerCommande(String commande) {
    String[] t = commande.split(":");
    if (t.length > 2) {
      int cont = Integer.parseInt(t[0]);
      int oper = Integer.parseInt(t[1]);
      int param = Integer.parseInt(t[2]);

      switch (cont) {

      case 0:
        if (oper == 0)
          haut(param);
        else
          bas(param);
        break;
```

```
      case 1:
        if (oper == 0)
          droite(param);
        else
          gauche(param);
        break;
      }
    }
  }
```

"MainActivity"

```
  private void lancerThreadClient() {
         Runnable r = new Runnable() {
            @Override
            public void run() {
                demmarerClient();
            }

        };
        Thread th = new Thread(r);
        th.start();
  }
  protected void demarrerClient() {
         try {
            Inet4Address i = (Inet4Address)
Inet4Address.getByName(edAdresse.getText().toString());
            s = new Socket(i, Integer.parseInt(edPort.getText().toString()));
            pw = new PrintWriter(new BufferedWriter(new OutputStreamWriter
                                              (s.getOutputStream())), true);
        } catch (UnknownHostException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
  }
  private void envoyer(final String cmd) {
        Runnable r = new Runnable() {
            @Override
            public void run() {
                if (pw != null) {
                    pw.println(cmd);
                }
            }
        };
        Thread th = new Thread(r);
        th.start();

    }

  private void haut() {
      envoyer("0:0:" + edAlt.getText().toString());
  }
  protected void bas() {
     envoyer("0:1:" + edAlt.getText().toString());
  }
  protected void droite() {
      envoyer("1:0:" + edAngle.getText().toString());
  }
  protected void gauche() {
     envoyer("1:1:" + edAngle.getText().toString());
  }
}
```