Mobile Computing

Practice # 3

Other Android Applications

1. Develop an application that calculates and presents the monthly payment and the total payment for a loan with a fixed interest rate. The user should provide the capital value (in €), the interest rate (in %) and the loan duration (in months). The calculations are simple and as follows:

```
temp = (1 + (interest_rate/100)/12) ^ duration

monthly_payment = capital * ((((interest_rate/100)/12) * temp) / (temp – 1))
```

Design a suitable user interface.

2. Develop an application to do money conversions from Euros to other currencies and also the inverse. The supported currencies and exchange ratios should be previously put on an XML file to be available on the xml folder of the Android resources. The XML file should have a root tag of Rates (<Rates> … </Rates>). Inside we have an arbitrary number of rows (tag Rate), each one having two elements: the currency code (3 letters and inside tag Currency) and the exchange rate Euro/Currency (inside tag Exchange). Some examples: AUD, CAD, GBP, JPY, RUB, USD, etc.

You can consult the real exchange rates, for instance in http://www.xe.com/ucc. Design an interface based on Spinners.

3. Many years ago (more than 20) a **text** based game called "Moon landing" recreated the moon landing of lunar module Eagle from Apollo 11. The purpose of the game was to achieve a perfect landing by burning fuel in the Eagle thruster.

The lunar module begins (elapsed = 0) stopped (velocity = 0) at an altitude of 15200 m and with 8165 kg of fuel. It has a thruster, activated by the astronaut, with a given power from 0 to 100% and during a brief interval indicated in seconds. When the thruster is activated an ascending force is created. The moon gravity is 1.625 m/s2. The application should show, after each thruster activation, the values of: altitude (in m), velocity (km/h), available fuel (kg), and elapsed time from the initial position (s). It should ask for the next thruster activation, reading from the player the power to use (in %) and the duration of the activation (in s). The new state of the lunar module should now be computed and shown to the player after the thruster activation period.

The player performs well if he achieves to reach the soil with a velocity less than 4 m/s (14.4 km/h). Otherwise the moon will acquire a new crater!

Implement this one player game with a suitable **text** interface (based on messages that are written in one or more interface controls). The needed computations are as follows:

```
// throttle and time: values from the player (% and s)

    thrust = throttle (%) * 1200.0
    consume = (thrust * time) / 2600.0
    fuel = fuel – consume
    avgmass = modmass + fuel + (consume / 2.0)        // modmass = 17198.0 (kg)
    force = thrust – (avgmass * gravity)              // gravity = 1.625 (m/s2)
    accel = force / avgmass
    vel1 = vel + (accel * time)                       //vel, vel1 and avgvel in m/s
    avgvel = (vel + vel1) / 2.0                       // velocity is < 0 if towards the soil
    alt1 = alt + (avgvel * time)                      // alt and alt1 in m
    elapsed = elapsed + time
```

```
if (alt1 <= 0.0)
    part = alt / (alt – alt1)
    vel = vel – ((vel – vel1) * part)
    alt = 0.0
    fuel = fuel + consume * (1 – part)
    elapsed = elapsed – time * (1 – part)
    if (vel >= -4.0)
        “OK”
    Else
        “Kaboom!”
Else
    alt = alt1
    vel = vel1
```