Q1)

Dalvik instructions are vastly different from Java instructions. DVM bytecode has 257 different instructions and 3 pseudo-instructions. Dalvik instructions are two to ten bytes long, and pseudo-instructions have a variable length.

Pseudo-instructions are used to store extra information related to other instructions (and thus are never executed).

Example:

i) add-int

Syntax:

add-int vx,vy,vz

Explanation:

Calculates vy+vz and puts the result into vx.

Example:

9000 0203 - add-int v0, v2, v3

Adds v3 to v2 and puts the result into v0.

ii)and-int

Syntax:

And-int vx, vy, vz

Explanation:

Calculates vy AND vz and puts the result into vx.

Example:

9503 0001- and-int v3, v0, v1

Calculates v0 AND v1 and puts the result into v3.

iii)move

Syntax:

move vx,vy

Explanation:

Moves the content of vy into vx. Both registers must be in the first 256 register range.

Example:

0110 – move v0, v1

Moves v1 into v0.

iv)return

Syntax:

return vx

Explanation:

Return with vx return value.

Example:

0F00 – return v0

Returns with return value in v0.

v)new-array

Syntax:

new-array vx,vy,type\_id

Explanation:

Generates a new array of type\_id type and vy element size and puts the reference to the array into vx.

Example:

2312 2500- new-array v2, v1, char[] // type@0025

Generates a new array of type@0025 type and v1 size and puts the reference to the new array into v2.

vi)fill-arrat-data

Syntax:

fill-array-data vx,array\_data\_offset

Explanation:

Fills the array referenced by vx with the static data. The location of the static data is the sum of the position of the current instruction and the offset.

Example:

2606 2500 0000 - fill-array-data v6, 00e6 // +0025

Fills the array referenced by v0 with the static data at current instruction+25H words location. The offset is expressed as a 32-bit number. The static data is stored in the following format:

0003 // Table type: static array data

0400 // Byte per array element (in this case, 4 byte integers)

0300 0000 // Number of elements in the table

0100 0000 // Element #0: integer 1

0200 0000 // Element #1: integer 2

0300 0000 // Element #2: integer3

vii)goto/16

Syntax:

goto/16 target

Explanation:

Unconditional jump by 16 bit offset2.

Example:

2900 0FFE - goto/16 002f // -01f1

Jumps to the current position-1F1H words. 002F is the label of the target instruction.

viii)int-to-long

Syntax:

int-to-long vx, vy

Explanation:

Converts the integer in vy into a long in vx,vx+1.

Example:

8106 - int-to-long v6, v0

Converts an integer in v0 into a long in v6,v7.

Q2)

Cloud computing, allows you to store your files and folders in a “cloud” area on the Internet, allowing you access to all of your files and folders wherever you are in the world – but you do need a physical device with Internet access to access it.

Mobile computing is taking a physical device with you. This could be a laptop or a mobile phone or some device which enables you to telework – working wherever you go because of the small size of the device you’re using.

Cloud computing - You can have all your files synchronised between devices so wherever you go, you’ll always have access to your files, but the technology doesn’t fully exist yet. It’s getting there, but it’s slow and temperamental, difficult to use and often the average user gets confused as to where the files are actually stored and/or where else they are stored. Even though you don’t need to carry round a laptop with you, you still need some physical device to access your service which almost defeats the point anyway. Having everything in your very own secure cloud so you can access anything anywhere is put back by the fact you have to find a computer to use anyway.

Mobile computing - The size of the mobile devices, keypad size and limited battery power make mobile computing not always feasible. Physical reconfiguration of the network because of moving between base stations, limited processing power, limited transmission power and low bandwidth also add to the disadvantages.

Q3)

Context

The word context is derived from the Latin “contextus”, which means “connection of words, coherence,” and from contexere “to weave together.”

No clear boundary divides what is and is not context.

Most interesting kinds of context are those that humans do not explicitly provide.

With advances in sensing and automated means of perceiving the physical environment, we can automatically collect much more implicit context.

Context helps us to do things better.

Context influences how we perceive information. Context enables us to manage the vast amount of information that surrounds us.

Context guides us through the information surrounding us. Context allows to discriminate “what is important” and“what is not”. Context helps us to adapt to our surroundings.

Example:

Few months ago google search for "who is anna?" would yield information about Anna Kournikova, famous Tennis player. But now Anna Hazare is shown. Ip based location, current activity on internet searches, page ranking, etc influence the search.

An example of a mobile app:

Friday is an app for Android phone that showcases context aware programming.

The idea being to provide a history of your communications, like calls, text messages, emails, etc, and combine those with other events your phone is able to record, like photos and battery drops, then mush it all together with data from third-party services, like Facebook. This is going to provide a personalised search engine for the user. Some contexts are explained this way, to justify it's context awareness:

a) This is an app that notifies user when the phone is running low on battery and there is a charger nearby.

b) Provides an alternate dialer app, which, rather than showing the recent calls log, shows the list of users you are most probably going to call based on your context

c)This is an app that shows you a daily travelogue, and activity graph.