**1.**

Design a VI to predict the trajectory of a cannonball shooting out of a canon on the ground. Draw a horizontal ground surface in the front panel with decoration elements. With four numerical controls, the initial velocity, the initial angle, percentage of energy loss when the ball hit the ground, and the percentage threshold of energy remained for the cannonball not bouncing again, try to draw the trajectory of the cannonball with a LED in the front panel and update the coordinate with property nodes. Please consider gravity but ignore the air drag effect.  
Try to tune the parameters to have the ball bounce 3 times before it stay on the ground, and make the final parameters as default values saved with the VI.

**2.**  
Apr 26 21:31:34 filterlog: 129,16777216,,1425041642,hn0,match,pass,in,4,0x0,,128,16110,0,none,17,udp,36,10.2.10.175,255.255.255.255,60221,3956,16  
Apr 26 21:31:34 filterlog: 129,16777216,,1425041642,hn0,match,pass,in,4,0x0,,128,4117,0,none,17,udp,78,10.2.10.1,208.93.4.206,137,137,58  
Apr 26 21:31:32 filterlog: 130,16777216,,1425041643,hn0,match,pass,in,4,0x0,,128,18168,0,DF,17,udp,351,10.2.10.10,10.2.23.123,53,59118,331  
Apr 26 21:31:32 filterlog: 129,16777216,,1425041642,hn0,match,pass,in,4,0x0,,128,10627,0,none,17,udp,82,10.2.10.10,140.112.2.2,59087,53,62  
Apr 26 21:31:32 filterlog: 129,16777216,,1425041642,hn0,match,pass,in,4,0x0,,128,10626,0,none,17,udp,98,10.2.10.10,140.112.2.2,60847,53,78  
Apr 26 21:38:10 filterlog: 105,16777216,,1425041627,hn1,match,block,in,4,0x2,0,109,9812,0,DF,6,tcp,48,207.46.13.171,10.2.10.15,25309,80,0,SEC,27892632,,8192,,mss;nop;nop;sackOK  
Apr 26 21:38:08 filterlog: 129,16777216,,1425041642,hn0,match,pass,in,4,0x0,,128,23124,0,DF,6,tcp,52,10.2.10.172,207.46.153.155,60590,443,0,S,2141064781,,8192,,mss;nop;wscale;nop;nop;sackOK  
Apr 26 21:38:08 filterlog: 130,16777216,,1425041643,hn0,match,block,in,4,0x2,0,128,8596,0,DF,6,tcp,48,10.2.10.2,10.2.102.10,41875,88,0,SEC,3256166066,,8192,,mss;nop;nop;sackOK  
Apr 26 21:38:07 filterlog: 130,16777216,,1425041643,hn0,match,pass,in,4,0x2,0,128,8592,0,DF,6,tcp,48,10.2.10.2,10.2.102.10,41874,88,0,SEC,2184502132,,8192,,mss;nop;nop;sackOK

The above text are derived from a section of firewall logs. Each line of the text above logged the important information of a packet being granted or blocked passing through the firewall.  
The fields of interest are explained with the example

Apr 26 21:38:10 filterlog: 105,16777216,,1425041627,hn1,match,block,in,4,0x2,0,109,9812,0,DF,6,tcp,48,207.46.13.171,10.2.10.15,25309,80,0,SEC,27892632,,8192,,mss;nop;nop;sackOK

Apr 26 21:38:10 : The date and time for the packet to be logged.  
hn1 : The interface of the firewall being logged. The field is always in the form of hnx, and x denotes the x-th network interface of the firewall.  
block : The packet is being rejected passing through the firewall. The other possible value of the field is pass, denoting the packet is being granted passing through the firewall.  
tcp : The internet protocol. Possible values includes tcp and udp for this field.  
48 : The length of the packet  
207.46.13.171: The source ip  
10.2.10.15: The target ip  
25309 : The source port.  
80 : The target port.

Please ignored all the other field unmentioned.  
Please copy and paste the above firewall logs into a string control in Labview. Separate a line with \n (or \n\r) and process a line in each loop with a while loop. End the while loop when the line retrieved is empty.  
Use regex to match above mentioned fields of interest in each loop (line). Output all the field of interest to a table (a 2D array of string indicator) to show the above logged information in the form of  
Date-time Interface-name Protocol Source-ip Source-port Target-ip Target-port Length Pass-or-block and save the result to a text file.

**3.**  
Please finish the on class example with state machine style VI to process a Boolean button cluster with “login”, “configure data acquisition”, and “data acquisition” buttons. Modify the VI to have the following extra functions.  
(1) A login popup window to request user enter username and password when the login button is pressed. Hardcode the username and password in the login subVI and send back information when correct login username and password is entered.  
(2) If the configure data acquisition or data acquisition buttons are pressed and the user is not previous login-ed, show the login popup window to request the user login first. If the user failed login, the user login window will popup again. If the user login successful, the program will continue the function corresponding to the button user pressed.  
(3) If the "configure data acquisition" button is pressed, show a centered window with two numerical inputs for sampling rate (samples/sec) and samples per acquisition, with default values of 1k samples/sec and 10000 samples per acquisition. The window dismissed after "OK" button is pressed.  
(4) If the "data acquisition" button is pressed, show a centered window with a waveform graph shows a sinusoidal waveform with the samples (horizontal axis) with the "samples per acquisition" parameter user set in "configure data acquisition". Convert the unit of horizontal axis to seconds with the "sampling rate" parameter user set in "configure data acquisition". The window dismissed after "OK" button is pressed  
(5) After login, if user press "configure data acquisition", proceed to "data acquisition" step after the configuration window is dismissed.

**4.**

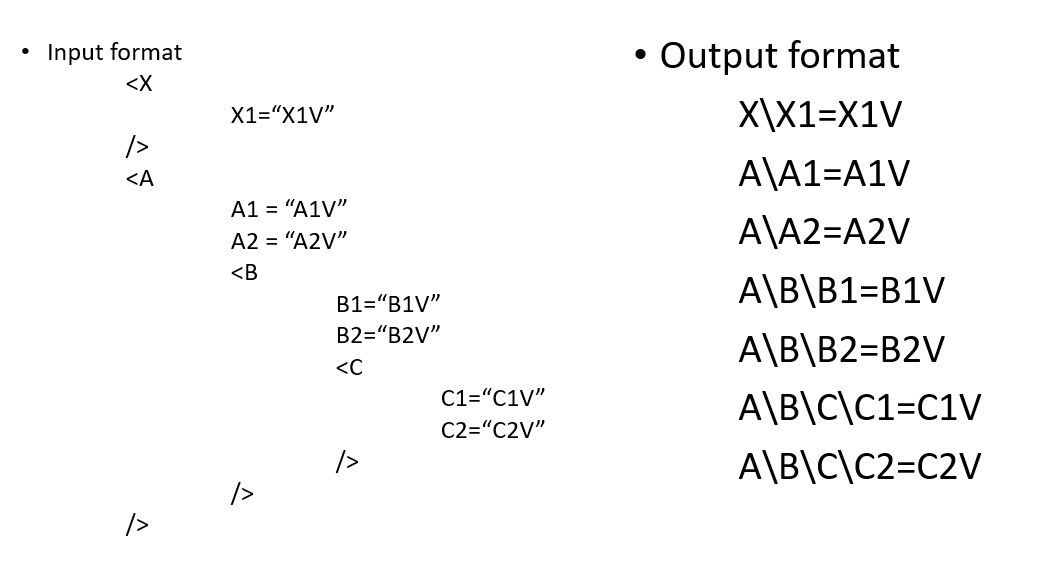
Write a **calculator** as the **Windows Calculator** program with basic “C +-\*/.= backspace“ function keys and 0~9 numeric keys. The calculator continuous runs until a “off” button was pressed.  
Please write the program using **“state machine”** architecture. (a separate VI). The +-\*/.= (map to enter key) and 0~9 keys can also be enter with physical keyboard

**5. Vending Machine**

* Write a VI to simulate the behaviors of a ticket vending machine with state-machine style programming.
* The vending machine can accept 5, 10, and 50 coins. With 3 buttons on the front panel to simulate the event of user insert 5, 10, and 50 coins into the vending machine.
* The ticket price is 35. A deposition display on the front panel shows the current deposition, and with a LED strobe for 1 second to simulate the dispensing event of a ticket when the deposition reaches 35.
* If the deposition is not zero after one ticket being dispensed. User can choose to deposite again to get another ticket or press return button to return the remaining deposition.

**6. Reformatting XML file**

* Please cut the sample.xml file session in last page and save to sample.xml as sample input.
* Write a xml file reformatting file VI with input file path, output file path, input String, and output Sting on the front panel. The VI will reading the input .xml file and reformatting the file into the specific format and saved to a .txt file
* PLEASE DO NOT USE THE BUILT-IN XML PARSER VI!!

****

**Sample.xml**

<ImageView

android:id="@+android:id/icon"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_gravity="center"

/>

<RelativeLayout

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginLeft="15dip"

android:layout\_marginRight="6dip"

android:layout\_marginTop="6dip"

android:layout\_marginBottom="6dip"

android:layout\_weight="1">

<TextView android:id="@+android:id/title"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:singleLine="true"

android:textAppearance="?android:attr/textAppearanceLarge"

android:ellipsize="marquee"

android:fadingEdge="horizontal" />

<TextView android:id="@+android:id/summary"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_below="@android:id/title"

android:layout\_alignLeft="@android:id/title"

android:textAppearance="?android:attr/textAppearanceSmall"

android:textColor="?android:attr/textColorSecondary"

android:maxLines="4" />

</RelativeLayout>