Review activity

For exam 2 (**suggested solution**)

The following is a list of exercises. Do the ones you feel less comfortable first so we can discuss in class if you have questions. These exercises do not include the term definitions you need to know for the exam, those that are described in the Study Guide as "Explain …" See the Study Guide for Exam 2 on the course Moodle site for reference.

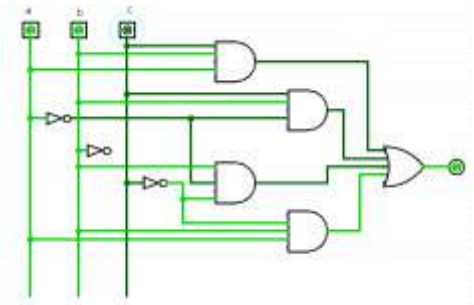
Form a group of 4-5, discuss all the problems together for the first 15 minutes. Then each student pick a few problems to work with. Ask questions each other while discussing and working on these problems.

1. Number base: Convert numbers between decimal and binary. Convert 23 to binary. Convert 10110 to decimal.
2. Truth table: Define a truth table to reflect the following facts.
   a. If today is not a Friday (**a**) and the weather is sunny (**b**), we will go to see a movie (**x**), if we don't have five friends present (**c**).
   b. If the number of friends (**c**) present is more than five, and the weather is sunny (**b**), we will go to see a movie (**x**), if the weekday is not a Friday.
   c. If today is Friday (**a**) and the weather is good (**b**), we will go to see a movie (**x**), regardless the number of friends present.
3. Derive the minterm expansion from the above truth table.
4. Build the circuits from the above minterm expansion.
5. Write an HMMM assembly function that adds even integers from 2 through 10, i.e., 2+4+…+10.
6. Write an HMMM assembly program that reads an integer **n** from keyboard, then determines if **n** is an even number or not. If not, add 1 to **n**, making it even. Call the above function to produce a sum. Print the result. You must use the proper **calln, jumpr, storer**, and **loadr** for the function call.
7. Solve the following problem using a Python **while** loop. Given a list of integers, e.g., [1, 5, 12, 14, 7, 6, 21], write a while loop to add all even numbers that are greater than or equal to 10. Print the sum.
8. Solve the same problem using a **for** loop by index.
9. Solve the same problem using a **for** loop by membership.
10. Create a 2D array of **n** by **m** with all members initialized to be 1.
11. Write a Python function that takes a 2D array and a **row** number as the parameters, return the specified row from the 2D array as a list.
12. Write a Python function that add all numbers of lower left portion of a 2D array, including the ones on the diagonal line.

1. Number base: Convert numbers between decimal and binary. Convert 23 to binary. Convert 10110 to decimal.
   23 = 16+4+2+1 = 10111, 10110 = 16+4+2 = 22
2. Truth table: Define a truth table to reflect the following facts.
   a. If today is not a Friday (**a**) and the weather is sunny (**b**), we will go to see a movie (**x**), if we don't have five friends present (**c**).
   b. If the number of friends (**c**) present is more than five, and the weather is sunny (**b**), we will go to see a movie (**x**), if the weekday is not a Friday.
   c. If today is Friday (**a**) and the weather is good (**b**), we will go to see a movie (**x**), regardless the number of friends present.

Friday (a)	Weather (b)	5-Friends (c)	Go movie? (x)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

3. Derive the minterm expansion from the above truth table.
   x = ~ab~c + ~abc + ab~c + abc
4. Build the circuits from the above minterm expansion.



5. Write an HMMM assembly function that adds even integers from 2 through 10, i.e., 2+4+…+10.

```
0 setn r1, 0    # s = 0
1 setn r3, -2   # r = -2
2 setn r2, 10   # upper limit
```

```
3 jltz r2, 8    # jumpt to end
4 add r1, r1, r2  # s = s + r2
5 add r2, r2, r3
6 jumpn 3       # continue loop
7 nop
8 write r1
9 halt
```

6. Write an HMMM assembly program that reads an integer **n** from keyboard, then determines if **n** is an even number or not. If not, add 1 to **n**, making it even. Call the above function to produce a sum. Print the result. You must use the proper **calln, jumpr, storer**, and **loadr** for the function call.

```
0  read r2
1  setn r8, 2    # for mod op
2  mod  r7, r2, r8 # r7 = r2 % 2
3  jeqzn r7, 5   # r2 is even, jump over adding
4  addn r2, 1    # make r2 even
5  setn r15, 50 # for stack
6  calln r14, 9 # call function
7  write r1      # return result in r1
8  halt
# function starts next line
9  storer r14, r15  # save return addr
10 setn r1, 0    # s = 0
11 setn r3, -2   # r = -2
12 nop           # r2 should have the upper limit
13 jltz r2, 18   # jumpt to end
14 add r1, r1, r2  # s = s + r2
15 add r2, r2, r3
16 jumpn 13      # continue loop
17 nop
18 nop
19 jumpr r14
```

7. Solve the following problem using a Python **while** loop. Given a list of integers, e.g., [1, 5, 12, 14, 7, 6, 21], write a while loop to add all even numbers that are greater than or equal to 10. Print the sum.
8. Solve the same problem using a **for** loop by index.
9. Solve the same problem using a **for** loop by membership.

```
def add_even_while(alist):
    s = 0
    i = 0
    while i < len(alist):
       if alist[i] % 2 == 0:
          s += alist[i]
       i += 1
    return s

def add_even_for(alist):
```

```
        s = 0
        for x in alist:
            if x % 2 == 0:
                s += x
        return s

    def add_even_fori(alist):
        s = 0
        for i in range(len(alist)):
            if alist[i] % 2 == 0:
                s += alist[i]
        return s

    a = [1,5,12,14,7,6,21]
    print(add_even_fori(a))
    print(add_even_for(a))
    print(add_even_while(a))
```

10. Create a 2D array of **n** by **m** with all members initialized to be 1.
11. Write a Python function that takes a 2D array and a **row** number as the parameters, return the specified row from the 2D array as a list.
12. Write a Python function that add all numbers of lower left portion of a 2D array, including the ones on the diagonal line.

```
from random import *
def p10(n, m):
    a = []
    for i in range(n):
        row = [1]*m
        a = a + [row]
    return a

def p11(m, row):
    return m[row]

def p12(m):
    s = 0
    for row in range(len(m)):
        for col in range(row + 1):
            s += m[row][col]
    return s

def random_v(m):
    for row in range(len(m)):
       for col in range(len(m[0])):
          m[row][col] = randint(1,10)
    return m
m = p10(3,4)
random_v(m)
print(m)
row = p11(m, 2)
```

```
print(row)
s = p12(m)
print(s)
```