German University in Cairo
Media Engineering and Technology Faculty
Prof. Dr. Slim Abdennadher
Dr. Aysha El Safty

January 19, 2017

CSEN102: Introduction to Computer Science
Winter Semester 2016-2017
Final Exam

Bar Code

Instructions: Read carefully before proceeding.

1) Please tick your major

	Major
	Engineering
	BI

2) Duration of the exam: 3 hours (180 minutes).
3) No books or other aids are permitted for this test.
4) This exam booklet contains 18 pages, including this one. Three extra sheets of scratch paper are attached and have to be kept attached. **Note that if one or more pages are missing, you will lose their points. Thus, you must check that your exam booklet is complete**.
5) Write your solutions in the space provided. If you need more space, write on the back of the sheet containing the problem or on the four extra sheets and make an arrow indicating that. **Scratch sheets will not be graded unless an arrow on the problem page indicates that the solution extends to the scratch sheets**.
6) When you are told that time is up, stop working on the test.

Good Luck!

Don't write anything below ;-)

Exercise	1	2	3	4	5	6	7	$\sum$
Possible Marks	18	10	4	23	14	11	20	100
Final Marks								

Exercise 1 General arithmetics (6+4+8=18 Marks)
Number conversions

In this exercise you have to convert numbers from one base to another. A number n of a base b other than 10 will be given on this page as

$$n_b$$

For decimal numbers, the subscript base can be omitted (*i. e.*, we write n instead of "n_{10}"). For bases b that are greater than 10, please use uppercase Latin letters (A, B, ...) as digits beyond 9, just as it is done with hexadecimal numbers.

a) Convert the following numbers into binary. Show your workout.

1. 762_8

   Solution:
   111110010_2

2. $B10A_{16}$

   Solution:
   1011000100001010_2

3. 432_5

   Solution:
   1110101_2

b) Convert the following and show your workout.

- CB_{15} into hexadecimal base (base 16) number

  Solution:
  BF_{16}

- 223_4 into hexadecimal base (base 16) number without converting the numbers into decimal

  Solution:
  $2B_{16}$

c) Give the least number of bits needed to perform the following operation in two's complement and perform the operation afterwards

$$-23 - 31$$

Solution:

1. 7 bits needed
2. $-23 = 1101001_2$
3. $-31 = 1100001_2$
4. $1101001 + 1100001 = 1001010$
5. 1001010 in 2's complement is $= -54_{10}$

Exercise 2 Binary representations (4+2+2+2=10 Marks)
Normalized scientific binary floating point

Recall the encoding schema for a normalized scientific binary floating point from Lecture 7:

Sign of mantissa 1 bit	Mantissa x bits	Sign of exponent 1 bit	Exponent y bits

a) We would like to store the decimal number -16.8125_{10} and be able to represent it in an accurate way. Your task is first to decide about the least values of x that corresponds to the size of the mantissa and y that corresponds to the size of the exponent (without the sign bits). Show your workout.

Solution:

$x = 9bits$
$y = 3bits$

b) Show the binary representation of the decimal number -16.8125_{10}

Solution:

$$-10000.1101$$

c) Show the binary number in normalized scientific notation.

Solution:

$$-.100001101 * 2^{+5}$$

d) Show how the binary number will be stored in the table below:

1	100001101	0	101
Sign of mantissa 1 bit	Mantissa x bits	Sign of exponent 1 bit	Exponent y bits

Exercise 3 Boolean Circuits (4 Marks)
Functionality

Given the following truth table

A	B	C	X_4	X_3	X_2	X_1	X_0
0	0	0	0	0	0	1	0
0	0	1	0	0	1	1	0
0	1	0	0	1	0	1	0
0	1	1	0	1	1	1	0
1	0	0	1	0	0	1	0
1	0	1	1	0	1	1	0
1	1	0	1	1	0	1	0
1	1	1	1	1	1	1	0

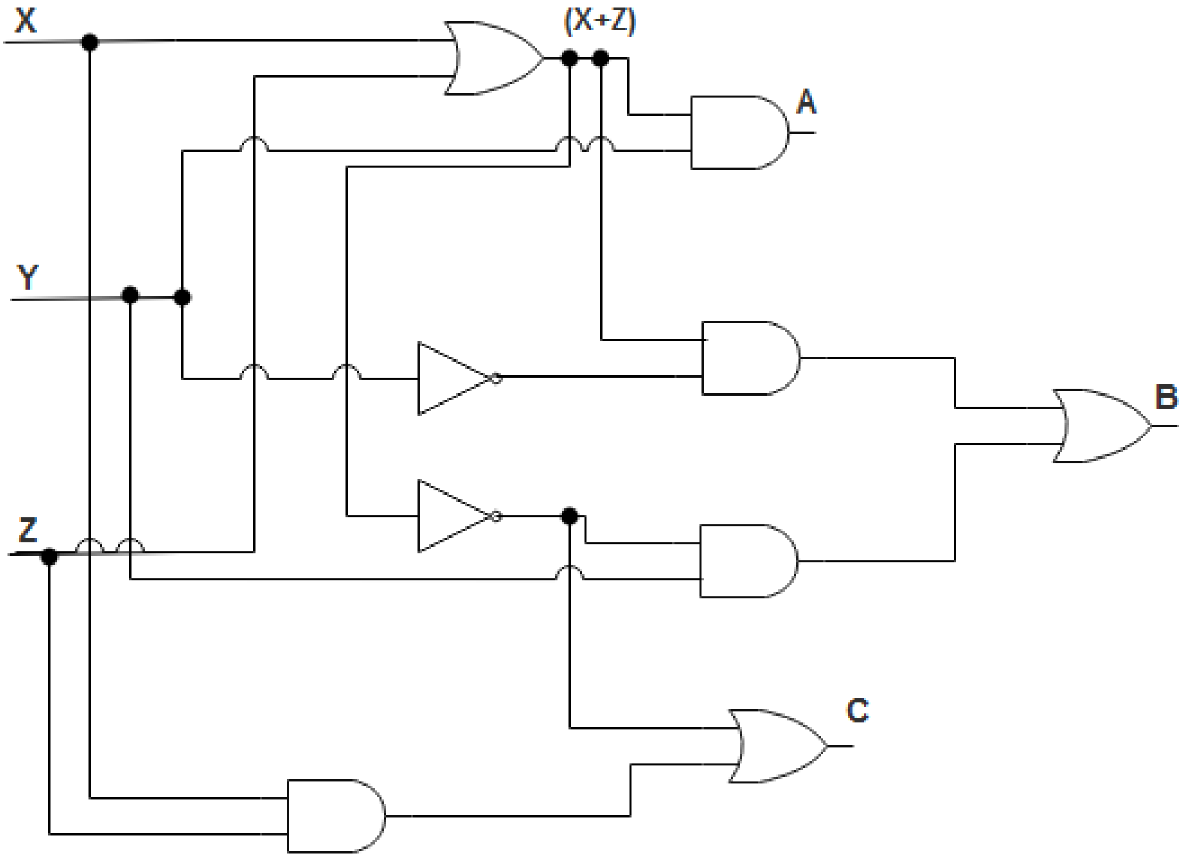
What is the functionality of the circuit where A, B, C are the input variables and X_4, X_3, X_2, X_1 and X_0 are the output variables? Justify your answer.

Solution:

$n * 4 + 2$

Exercise 4 Boolean functions (3+4+4+6+2+4=23 Marks)
Circuit improvement

Given the following circuit:



a) Extract the three Boolean expressions for A, B and C from the circuit.

Solution:

$$A = y(x + z)$$
$$B = y(x + z)' + y'(x + z)$$
$$C = xz + (x + z)'$$

b) Draw a truth table for the circuit.

Solution:

X	Y	Z	A	B	C
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	0	1	0
1	0	1	0	1	1
1	1	0	1	0	0
1	1	1	1	0	1

c) Use the sum-of-products algorithm to determine the Boolean expressions that correspond to the truth table.

Solution:

$$A = x'yz + xyz' + xyz$$
$$B = x'y'z + x'yz' + xy'z' + xy'z$$
$$C = x'y'z' + x'yz' + xy'z + xyz$$

d) Simplify the Boolean expressions received from part **(c)** to reach the expressions in part **(a)**. Use the axioms of the Boolean Algebra. Please mention the applied rules.

$x + 0 = x$	$x * 1 = x$	
$x + 1 = 1$	$x * 0 = 0$	
$x + x = x$	$x * x = x$	
$x + x' = 1$	$x * x' = 0$	
$(x')' = x$		
$x + y = y + x$	$xy = yx$	Commutativity
$x + (y + z) = (x + y) + z$	$x(yz) = (xy)z$	Associativity
$x(y + z) = xy + xz$	$x + yz = (x + y)(x + z)$	Distributivity
$(x + y)' = x'y'$	$(xy)' = x' + y'$	DeMorgan's Law

Solution:

The sum of products algorithm yields the following conjuncts:

$$A = x'yz + xyz' + xyz$$

$$\begin{aligned}
f(x, y, z) &= x'yz + xyz' + xyz \\
&= x'yz + xy(z' + z) && \text{(Distributivity)} \\
&= x'yz + xy && (x + x' = 1) \\
&= yx'z + yx && \text{(Commutativity)} \\
&= y(x'z + x) && \text{(Distributivity)} \\
&= y((x' + x)(z + x)) && \text{(Distributivity)} \\
&= y(z + x) && (x + x' = 1)
\end{aligned}$$

$$B = x'y'z + x'yz' + xy'z' + xy'z$$

$$\begin{aligned}
f(x, y, z) &= x'y'z + x'yz' + xy'z' + xy'z \\
&= x'y'z + x'yz' + xy'(z' + z) && \text{(Distributivity)} \\
&= x'y'z + x'yz' + xy' && (x + x' = 1) \\
&= x'y'z + xy' + x'yz' && \text{(Commutativity)} \\
&= y'x'z + y'x + x'yz' && \text{(Commutativity)} \\
&= y'(x'z + x) + x'yz' && \text{(Distributivity)} \\
&= y'((x' + x)(z + x)) + x'yz' && \text{(Distributivity)} \\
&= y'(z + x) + x'yz' && (x + x' = 1) \\
&= y'(z + x) + yx'z' && \text{(Commutativity)} \\
&= y'(z + x) + y(x + z)' && \text{(DeMorgan's Law)}
\end{aligned}$$

$$C = x'y'z' + x'yz' + xy'z + xyz$$

$$
\begin{aligned}
f(x,y,z) &= x'y'z' + x'yz' + xy'z + xyz & \\
&= x'z'y' + x'z'y + xzy' + xzy & \text{(Commutativity)} \\
&= x'z'(y'+y) + xz(y'+y) & \text{(Distributivity)} \\
&= x'z' + xz & (x + x' = 1) \\
&= (x+z)' + xz & \text{(DeMorgan's Law)}
\end{aligned}
$$

e) Can you simplify the expressions further? If yes, apply the simplification rules and simplify.

Solution:

No, no further simplification.

f) What is the functionality of the circuit?

Solution:

If the input number is less than 4 the circuit adds 1 to the input, otherwise it subtracts 2 from the input.

Exercise 5 Boolean Circuits (1+3+10=14 Marks)
MultiplyByThree

Assume we have a number X consisting of 4 bits and we would like to design a *MultiplyByThree* circuit that performs the operation $3X$.

a) How many input variables will the circuit have?

Solution:

4 variables

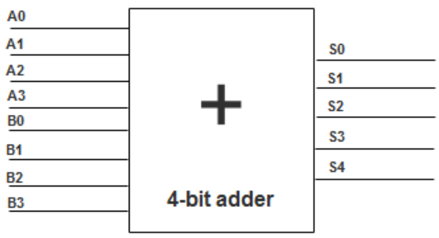
b) How many output variables will the circuit have? Justify your answer.

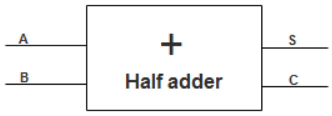
Solution:

6 variables

c) Design a *MultiplyByThree* circuit using one 4-bit adder. A 4-bit adder is a circuit that performs the addition of two number consisting of 4 bits each. Additionally, you have a half adder, it is a circuit that performs the addition of two numbers consisting of one bit each. Assume that we have already manufactured 4-bit adder and half adder shown below.

Note: This question can be solved using one 4-bit adder only.





Solution:

Exercise 6 Algorithms and efficiency (4+2+1+4=11 Marks)
Mysterious

Given the following program

```
l = eval(input())
n = len(l)
x = n
i = 0
total  = (x+1)*(x+2)/2
while(i<n):
  total -= l[i]
  i += 1
print(int(total))
```

a) What is the output of the program for the following input

$$l = [1, 2, 4, 6, 3, 7, 8]$$

Trace your program.

Solution:

n	i	$total$
7	0	36.0
7	1	35.0
7	2	33.0
7	3	29.0
7	4	23.0
7	5	20.0
7	6	13.0
7	7	5.0

```
Output = 5
```

b) Give the total number of operations for the execution of the program

```
l = eval(input()) ----------> 1 operation executed once
n = len(l) ----------------->
x = n ---------------------->
i = 0 ---------------------->
total  = (x+1)*(x+2)/2  ---->
while(i<n): ---------------->
  total -= l[i] ------------>
  i += 1-------------------->
print(int(total))----------->
```

Solution:

$3 * n + 7$

c) Give the order of magnitude of the program above.

Solution:

$O(n)$

d) What is the functionality of the program for any list of n integers and these integers are in the range of 1 to $n + 1$, having no duplicates (i.e. all elements in the list are different).

Solution:

The algorithm prints the missing element in the list having elements from 1 to $n + 1$.

Exercise 7 Iterative Algorithms (20 Marks)

Hint: In this question, take some time to read the whole question twice and the example then start to solve.

You are supposed to implement a game, where one of the players is the computer and the other is a human.

We are having three bags $1, 2, 3$. Each player takes turn and removes between 1 and 5 objects from one of three bags. The player who removes the last object(s) wins the game. Here are the rules:

- Each bag contains 10 objects in the beginning of the game.
- The first turn is for the human then comes the computer's turn.
- The player can remove object(s) from a single bag at a time
- The player can not choose to remove 0 objects.
- The number of objects has to be between 1 and 5 (inclusive).
- The player has to enter the bag first (1 or 2 or 3) and then the number of object(s) that have to be removed from that bag.
- You have also to check whether the human's input is valid. Is it a valid bag (1 or 2 or 3)? Is it a valid number of objects (between 1 and 5)? Also are there enough objects in the bag? If any of these conditions are not met, then the user should be asked for an input again (bag and number of objects).
- Note that you need to make sure that the computer's selection should be always valid. The computer's turn should not end unless it guessed a correct bag (containing at least 1 object) and a valid number of objects from that bag. Once the computer guesses the correct bag and number of objects the human has the turn to play. **Hint:** use the function `random.randint()`
- The computer should never guess an invalid bag nor invalid number of objects to be removed.
- The game ends when all bags are empty.
- The player who removes the last object wins the game.

Note: Your program will roughly consist of two parts; one part dealing with the human's turn and one part dealing with the computer's turn. Note that the turns have to repeat until all bags are empty.

Here is an example run of the program. Your algorithm should handle and display the following:

```
10 10 10
Select a bag: 3
Select number of objects: 4
You took: 4  objects from bag: 3
10 10 6

Computer took: 2  objects from bag: 3
10 10 4

Select a bag: 5
Select number of objects: 3
Please select a correct bag.

Select a bag: 3
Select number of objects: 4
You took: 4  objects from bag: 3
10 10 0

Computer took: 4  objects from bag: 2
10 6 0
```

```
Select a bag: 2
Select number of objects: 6
Please enter a valid number.

Select a bag: 2
Select number of objects: 5
You took: 5  objects from bag: 2
10 1 0

Computer took: 1  objects from bag: 2
10 0 0

Select a bag: 2
Select number of objects: 6
Please enter a valid number.

Select a bag: 2
Select number of objects: 3
You cannot remove this number.

Select a bag: 1
Select number of objects: 6
Please enter a valid number.

Select a bag: 1
Select number of objects: 3
You took: 3  objects from bag: 1
7 0 0

Computer took: 1  objects from bag: 1
6 0 0

Select a bag: 1
Select number of objects: 2
You took: 2  objects from bag: 1
4 0 0

Computer took: 2  objects from bag: 1
2 0 0

Select a bag: 1
Select number of objects: 2
You took: 2  objects from bag: 1
0 0 0

Congratulations, You won
Done
```

Exercise 7 Solution:

Solution:

```
import random

N = [10,10,10]
print(N[0], N[1], N[2])
A = True
B = True
C = True
turn = 'p'
while (A or B or C):
  if (turn == 'p'):
    bag = eval(input("Select a bag"))
    p = eval(input("Select number of objects"))
    if(bag !=1 and bag !=2 and bag !=3):
      print("Please select a correct bag")
    elif(p < 1 or p >5):
      print("Please enter a valid number")
    elif(bag == 1 and p > N[0] or bag == 2 and p > N[1] or bag == 3 and p > N[2]):
      print("You cannot remove this number")
    else:
    #  print("hey")
      if(bag == 1):
        N[0] -= p
      elif(bag == 2):
        N[1] -= p
      elif(bag == 3):
        N[2] -= p
      print("You took:",p," objects from:",bag)
      print(N[0], N[1], N[2])
      turn = 'c'
  if (turn == 'c'):
   #   print("hello")
      cBag = random.randint(1,3)
      if(N[cBag-1]>=1 and turn == 'c'):
        if (N[cBag-1] <=5):
          comp = random.randint(1,N[cBag-1])
        else:
          comp = random.randint(1,5)

        N[cBag-1] -= comp
        print("Computer took:",comp," objects from:", cBag)
        print(N[0], N[1], N[2])
        turn = 'p'
      elif(N[cBag-1]==0 and turn == 'c'):
        cBag = random.randint(1,3)

  if (N[0] == 0):
        A = False
       # print("First bag is empty")
  if (N[1] == 0):
        B = False
     #   print("Second is empty")
  if (N[2] == 0):
        C = False
```

```
    #  print("Third is empty")

if(turn == 'p'):
  print(" Computer wins")
else:
  print(" Congratulations, You won")

print("Done")
```

Another Solution:

```
import random
bag1,bag2,bag3 = 10,10,10
turn = True
game_over = False
while(not(game_over)):
  valid = True
  if(turn):
    #PLAYER'S TURN
    bag = eval(input("select a bag"))
    remove = eval(input("select number of objects"))
    if(bag > 3 or bag < 1):
      print("please select a correct bag")
      valid = False
    if(remove == 0):
      print("please enter a valid number")
      valid = False
    if(remove > 5 or remove < 1):
      print("please enter a valid number")
      valid = False
    if((bag == 1 and remove > bag1) or (bag == 2 and remove > bag2) or (bag == 3 and
      print("you can not select this number")
      valid = False
  else:
    # COMPUTER'S TURN
    bag = random.randint(1,3)
    if((bag == 1 and bag1 ==0) or (bag == 2 and bag2 ==0) or (bag == 3 and bag3 ==0))
      valid = False
    if(valid):
      min = 6
      if(bag == 1):
        if(bag1 < 5):
          min = bag1
      if(bag == 2):
        if(bag2 < 5):
          min = bag2
      if(bag == 3):
        if(bag3 < 5):
          min = bag3
      if(min >= 5):
        min = 5
      remove = random.randint(1,min)
  # TAKING ACTIONS IF IT WAS A VALID MOVE
  if(valid):
    if(bag == 1):
```

```
      bag1 -= remove
    if(bag == 2):
      bag2 -= remove
    if(bag == 3):
      bag3 -= remove
    if(turn):
      player = "you"
    else:
      player = "computer"
    print(player,"took:" , remove , "objects from bag :" , bag)
    print(bag1 , bag2 , bag3)
    if(bag1 == 0 and bag2 == 0 and bag3 == 0):
      game_over = True
    turn = not(turn)
if(turn):
  print("Computer wins")
else:
  print("You win")
```

Scratch paper

Scratch paper

Scratch paper