Practice Sheet

Problem 1 *Properties of Relations*

Suppose that R and S are equivalence relations on a set A. Prove or refute by a counter example each of these statements:

1. $R \cap S$ is an equivalence relation
2. $R \cup S$ in an equivalence relation
3. $S \circ R$ is irreflexive
4. $R \setminus S$ is an equivalence relation

Solution:

1. $R \cap S$ is an equivalence relation as:
   (a) $(a, a) \forall a \in A$
   (b) if $(a, b) \in (R \cap S)$ then $(b, a) \in (R \cap S)$ as both R and S are symmetric
   (c) if $\{(a, b), (b, c)\} \in (R \cap S)$ then $(a, c) \in (R \cap S)$ as both R and S are transitive
2. $R \cup S$ is not an equivalence relation since
   $A = \{a, b, c\}, R = \{(a, b), (b, a), (a, a), (b, b), (c, c)\}, S = \{(b, c), (c, b), (a, a), (b, b), (c, c)\}$
   $\{(a, b), (b, c)\} \in (R \cup S)$ but $(a, c) \notin (R \cup S)$
3. $S \circ R$ is not irreflexive: $A = \{a\}, R = \{(a, a)\}, S = \{(a, a)\}$
   $(a, a) \in (S \circ R)$
4. $R \setminus S$ is not an equivalence relation since
   $\forall a \in A(a, a) \notin (R \setminus S)$

Problem 2 *Function injectivity/surjectivity*

1. Prove or refute that the function $f : \mathbb{N} \to \mathbb{N}; n \mapsto 2n + 1$ is bijective.
2. Prove or refute that the function $g : \mathbb{N} \to \mathbb{N}; n \mapsto n + 1$ is bijective.

Solution:

1. f is not bijective it is not surjective, For example $2 \in \mathbb{N}$ and there is no $n \in \mathbb{N}$ that $2 = 2n+1$.
2. g is bijective: Let $n \in (\mathbb{N} \setminus \{0\})$, then $n = g(n - 1)$, so g is surjective. It is also injective by the third Peano axiom.

Problem 3 *Proving*

1. Using mathematical induction, prove that for all positive integers n, we have $1^3 + 2^3 + 3^3 + ... + n^3 = n^2(n + 1)^2/4$
2. Proof using the contrapositive that for all integers n, if we have $7n + 9$ is even then n is odd.

Solution:

1. Base Case: Trivial
   Assuming the above statement to be true with n = k.
   Step Case: $1^3+2^3+3^3+...+k^3+(k+1)^3 = n^2(n+1)^2/4+(k+1)^3 = (k+1)^2[k^2/4+(k+1)] = (k + 1)^2[k^2 + 4k + 4]/4 = (k + 1)^2[(k + 2)^2]/4$
2. Suppose n is not odd.
   Thus n is even, so $n = 2a$ for some integer a.
   Then $n + 9 = 7(2a) + 9 = 14a + 8 + 1 = 2(7a + 4) + 1$. Therefore $7n + 9 = 2b + 1$, where b is the integer $7a + 4$.
   Consequently $7n + 9$ is odd. Therefore $7n + 9$ is not even.

Problem 4 *CNF/DNF*

Write the CNF and DNF of the boolean function that corresponds to the truth table below.

x_1	x_2	x_3	f
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Solution:
DNF: $\overline{x_1 x_2 x_3} + \overline{x_1} x_2 \overline{x_3} + x_1 \overline{x_2 x_3} + x_1 \overline{x_2} x_3 + x_1 x_2 \overline{x_3}$
CNF: $(x_1 + x_2 + \overline{x_3})(x_1 + \overline{x_2} + \overline{x_3})(\overline{x_1} + \overline{x_2} + \overline{x_3})$

Problem 5 *Converting*

Convert the following values to their decimal equivalent

1. 10101_2
2. 75071_8
3. 12201_3
4. 26501_7
5. 430021_5

Solution:

1. 21
2. 31289
3. 154
4. 7106
5. 14386

Problem 6 *More Converting*

Convert the following values to their binary equivalent

1. 4072_{10}
2. CDA_{16}

Solution:

1. 111111101000
2. 110011011010

Problem 7 *b-complement*

Consider a fixed size integer numeral system using the b-complement notation for negative integers with the base b=7 and n=4 digits

1. What is the smallest number that can be represented in the number system?
2. What is the largest number that can be represented in the number system?
3. What is the representation of 68_{10} in the numeral system?
4. What is the representation of -68_{10} in the numeral system?

Note:
The range of a b-complement number system with base b and n digits is given by $[-\frac{1}{2}b^n, \frac{1}{2}b^n - 1]$
Solution:

b = 7 and n = 4

1. $-\frac{1}{2}b^n = -\frac{1}{2}7^5 = -1200$
2. $\frac{1}{2}b^n - 1 = \frac{1}{2}7^4 - 1 = 1199$
3. $68_{10} = 1 \cdot 7^2 + 2 \cdot 7^1 + 5 \cdot 7^0 = 125_7$
4. We already know that $68_{10} = 0125_7$. Calculating $a_i' = (b-1) - a_i$ for all digits a_i and adding 1 to it, we obtain $-68_{10} = 6541_7 + 1_7 = 6542_7$

Problem 8 *IEEE 754*

Compute the IEEE 754 single precision binary representation of the following numbers:

1. 0.71875
2. 27.3515625

Solution:

1. $23/32 = 0.71875$
   This is a positive fraction so:
   Sign bit = 0
   $0.71875/2^{-1} = 1.4375 \Rightarrow$ exponent = 127 - 1 = 126
   126 in binary is 01111110
   0.4375 = $2^{-2} + 2^{-3} + 2^{-4} \Rightarrow$ 01110000000000000000000
   Result = 0 01111110 01110000000000000000000
2. 27.3515625 is positive so Sign bit = 0
   $27.3515625/2^4 = 1.70947265625 \Rightarrow$ exponent = 127 + 4 = 131
   131 in binary is 10000011
   0.70947265625 = $2^{-1} + 2^{-3} + 2^{-4} + 2^{-6} + 2^{-8} + 2^{-9} + 2^{-11} \Rightarrow$ 10110101101000000000000
   Result = 0 10000011 10110101101000000000000

Problem 9 *QMC*

Execute the Quine-McCluskey algorithm to get the minimum polynomial for the Boolean function given by

x_1	x_2	x_3	x_4	f
1	1	1	1	1
1	1	1	0	0
1	1	0	1	1
1	1	0	0	0
1	0	1	1	1
1	0	1	0	0
1	0	0	1	1
1	0	0	0	0
0	1	1	1	1
0	1	1	0	0
0	1	0	1	0
0	1	0	0	0
0	0	1	1	1
0	0	1	0	0
0	0	0	1	0
0	0	0	0	0

Solution:

QMC_1 :

x_1	x_2	x_3	x_4
T	T	T	T
T	T	F	T
T	F	T	T
T	F	F	T
F	T	T	T
F	F	T	T

x_1	x_2	x_3	x_4
T	T	X	T
T	X	T	T
X	T	T	T
T	X	F	T
T	F	X	T
X	F	T	T
F	X	T	T

x_1	x_2	x_3	x_4
T	X	X	T
X	X	T	T

Therefore the prime implicants are $x_1 x_4$ and $x_3 x_4$

QMC_2 :

	TTTT	TTFT	TFTT	TFFT	FTTT	FFTT
$x_1 x_4$	T	T	T	T	F	F
$x_3 x_4$	T	F	T	F	T	T

Therefore both prime implicants are essential.

Final result: $f = x_1 x_4 + x_3 x_4$

Problem 10 *Universal function*

Prove that not-or (↓) is an universal function, by showing that it is sufficient to express all possible boolean functions.

Solution:

$\neg X = X \downarrow X$

X	¬ X	$X \downarrow X$
T	F	F
F	T	T

$X \wedge Y = (X \downarrow X) \downarrow (Y \downarrow Y)$

X	Y	$(X \downarrow X) \downarrow (Y \downarrow Y)$
T	T	T
T	F	F
F	T	F
F	F	F

$X \vee Y = (X \downarrow Y) \downarrow (X \downarrow Y)$

X	Y	$(X \downarrow Y) \downarrow (X \downarrow Y)$
T	T	T
T	F	T
F	T	T
F	F	F

Problem 11 *fork()*

How many processes are created when running the following C program ?

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main(void) {
    pid_t pid = fork();  // fork #1
    pid = fork();        // fork #2
    pid = fork();        // fork #3
    if (pid == 0) {
      fork();            // fork #4
    }
    fork();              // fork #5
    return 0;
}
```

Solution:

1. Creates an additional processes. You now have two processes.
2. is executed by two processes, creating two processes, for a total of four.
3. is executed by four processes, creating four processes, for a total of eight. Half of those have pid==0 and half have pid != 0
4. is executed by half of the processes created by fork 3 (so, four of them). This creates four additional processes. You now have twelve processes.
5. is executed by all twelve of the remaining processes, creating twelve more processes; you now have twenty-four.

Problem 12 *Assembly*

Translate the following C code into assembly. Use the instruction set of the lecture notes. Assume a has memory address 12 and b has memory address 13.

```
int b = 0;
for (int a = 0; a != 10; a++) {
    b = b + a
}
```

Solution:

```
// assuming a and b lie in memory addresses 12 and 13
0   LOAD #0      001 1 0000
1   STORE 13     010 0 1101 // int b = 0;
2   STORE 12         010 0 1100 // int a = 0;

3   LOAD 12      //load a into memory
4   EQUAL #10    101 1 1010 // next instruction is skipped when the value in a is 10,
5   JUMP #7      110 1 1000 // if a != 10 jump to instruction 7
6   HALT         111 0 0000


7   ADD #1       011 1 0001 // a = a + 1;
8   LOAD 13      001 0 1101 // load b into accumulator
9   ADD 12       011 0 1100 // accumulator = b + a;
10  STORE 13     010 0 1101 //store accumulator value back to b
11  JUMP #3
```

Problem 13 *isPalindrome*

Write a Haskell program that can determine whether a list is a palindrome. Examples below.

```
*Main> isPalindrome [1,2,3]
False
*Main> isPalindrome "madamimadam"
True
*Main> isPalindrome [1,2,4,8,16,8,4,2,1]
True
```

Solution:

```
isPalindrome :: (Eq a) => [a] -> Bool
isPalindrome xs = xs == (reverse xs)
isPalindrome' []  = True
isPalindrome' [_] = True
isPalindrome' xs  = (head xs) == (last xs) && (isPalindrome' $ init $ tail xs)
```

Problem 14 *Drop Every N'th element*

Write a Haskell program that removes every N'th element from a list. Example below.

```
*Main> dropEvery "abcdefghik" 3
"abdeghk"
```

Solution:

```
dropEvery :: [a] -> Int -> [a]
dropEvery list count = helper list count count
  where helper [] _ _ = []
        helper (x:xs) count 1 = helper xs count count
        helper (x:xs) count n = x : (helper xs count (n - 1))
```