CPSC 201: Introduction to Computer Science
Carsten Schürmann
Date: February 26, 2002

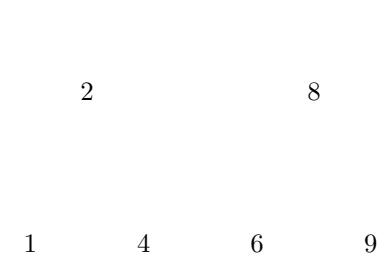
Midterm

Friday, March 1, 2002.

Name:	
Problem 1:	
Problem 2:	
Problem 3:	
Sum:	

Problem 1: Programming in ML

Assume that we are given a binary search tree. A binary search tree



with the representation invariant that any tree of the form `Node (left, x, right)` satisfi It has the following form.

Question 1: Write

1. What is the type of this function?
2. What is the invariant of this function?
3. Write the function itself.

Problem 3: Boolean Logic (40 Points)

A majority function $M_n(x_1, \ldots, x_n)$ is *true* if a majority of its inputs are *true* and $false$ otherwise (assume n odd). In this assignment we study majority functions with *three* inputs.

1. Write the truth table for a majority function $M_3(x_1, x_2, x_3)$ with just three inputs.
2. Write the Boolean expression for this majority function.
3. Construct the circuit for $M_3(x1, x2, x3)$ using only AND and OR gates.
   Hint: A typical implementation uses three AND gates and three OR gates. Reformulate your answer from 2. using distributivity laws.
4. Does $M_3(x_1, x_2, x_3)$ and NOT gate form a complete Boolean basis? Justify your answer.
   Hint: A NAND Gate alone, a NOR Gate alone, or the set of AND, OR and NOT gate form a complete Boolean basis. Try to implement one of these (or possibly other basis) using the $M_3(x_1, x_2, x_3)$ and a NOT gate.

Problem 3: Programming Machine Code

`LOAD` A R_i: Load the contents of memory at address A into register R_i.

`LOADI` R_i R_j : Load the contents of memory at the address contained in register R_i into register R_j.

`MOVE` R_i R_j: Move the contents of register R_i to register R_j.

`STORE` R_i A: Store the contents of register R_i into memory at address A.

`ADD` R_i R_j: Add contents of registers R_i and R_j and store the result in register R_0.

`MULT` R_i R_j: Multiply contents of registers R_i and R_j and store the result in register R_0.

`CONST` C R_i: Moves constant C to register R_i.

`JMP` A: Jump to memory address A and continue program execution from that location.

`CJMP` A: Jump to memory address A iff register R_0 is zero.

`OUT` R_i: Output contents of register R_i.

`HALT`: Halt the program.

Hints and Assumptions:

- Assume that memory has 1000 locations from address 0 to address 999. You might want to divide the memory into two portions, one to store the program and other to store the data.
- Each instructions requires just a single memory address.