CS 270 Assignment 3 (Conjunctive Normal Form)

Due Thur Nov. 16 at 9am. No late assignments will be accepted.

Name 1:____

Drexel Username 1:____

Instructions: This assignment is to be done using DrRacket and is submitted in BBLearn.

Download, from the course website or piazza, the file assign3.rkt. This file is to be loaded into DrRacket and will be modified as you work on your solution. Rename the file assign3sol.rkt. When you are done you will submit the file assign3sol.rkt, which contains your solution, along with the file readme.txt, which should contain a brief summary of your solution along and any difficulties you encountered. Please indicate any questions you did not complete and roughly how long it took you to complete this part of the assignment.

Assignment Overview

In this assignment you create a function that converts an arbitrary boolean expression into an equivalent one in conjunctive normal form (CNF). Recall that CNF is a conjunction (and) of clauses which are disjunctions (or) of literals. A literal is either a variable or a negated variable. Note that a single clause is in CNF as is a single literal or constant.

Conversion to conjunctive normal form is done in two steps:

1) [nnf] convert to negative normal form using DeMorgan's law

2) [cnf] convert expressions in negative normal form to conjunctive normal form using the distributive law.

For the function cnf to be correct the output must be in CNF and it must be equivalent to the input. Separate functions nnf? and cnf? are written to check that the output from nnf and cnf is in negative normal form or conjunctive normal form respectively. A function is-equivalent?, which uses the tautology checker from Lab 5, will be written to check if two boolean expressions are equivalent.

There are four parts to the assignment. In Part 0 code from Lab 5 is provided so that you can easily use the functions from there as helper functions in the assignment. Code for (1) is provided in Part I. Students should study this code before writing code for (2) in Part II. In Part II students write cnf? and the function cnf along with helper functions that guide them through the implementation of cnf? and cnf. In Part III students write a function is-equivalent? to test of two boolean expressions are equivalent.

The goal of this assignment is to further develop recursive programming skills and to reason informally about recursive programs in addition to providing informal and formal (using Racket itself) specifications. In a follow up assignment students will reason formally about these recursive functions.