when writing a class, you are not writing a program
- no static method

Accessor Methods
Access, dont change
public int getWidth(){ returns value of variable
return width; - width height private
} dont want user to change

gives client access to the data stored in private va.

mutator method
Alters the values of the data members
public void grow(int width, int height) {
this.width = width;
this.height = height;

instance methods

setter methods:
responsible for preventing inappop. changes
public void setWidth(int w){
if (w<=0) {
throw new IllegalArgumentException()
}
this.width = w;
public void setHeight(int h){
if (h<=0) {
throw
}
this.height = h;
public void grow (int dw, int dh){
this.setWidth(width+dw)
this.setHeight(height+dh) only one parameter but 2 things passed
} "this"

mutator that scales dim. by a factor
public void scale (int factor) ←
Accessor that determines if it is square
public boolean isSquare() ← passed address of object called on (this)

CS131 Problemset 2
Name: Emily Opresnick
Collaborator: None

Problem 1:

a. zybook 2 3.1d

False, while true that every boolean expression can be equivalent to some CNF formula and/or a DNF formula, this does not mean the boolean expression given has to be a CNF or DNF

b. zybook 2.3.2i

$x+\bar{y}+z$ is a DNF and CNF

c. zybook 2.3.3e

$(\overline{xy+z})(x+z)$

$(\overline{xy}\ \overline{z})(x+z)$ Demorgans $\overline{A+B} = \bar{A}\cdot\bar{B}$

$(\overline{xy})\,\bar{z}\,(x+z)$ Associative

$(\overline{xy})(x\bar{z}+z\bar{z})$ Distributive

$(\overline{xy})(x\bar{z}+0)$ Complement

$(\overline{xy})(x\bar{z})$ Identity

$xy+\bar{x}+z$ Demorgans

d. zybook 2.5.4a

a. $\bar{x}+y$ $\quad x\uparrow y = \overline{xy}$ $\quad \overline{xy} = \overline{\bar{x}+y}$

NAND:

$\bar{x}+y = \overline{x\bar{y}}$ Demorgans $= x\uparrow\bar{y}$

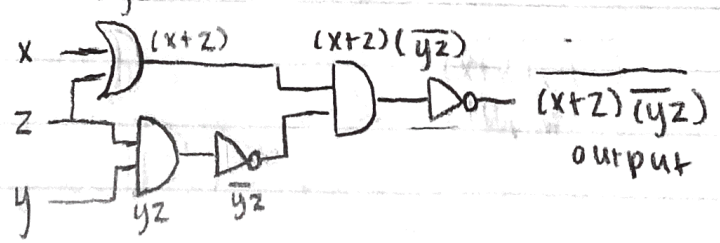
NOR

$\bar{x}+y = \overline{\bar{x}\downarrow y}$

$x+y = \overline{x\downarrow y}$

e. zybook 2.6.1c

$\overline{(x+z)(\bar{y}z)}$



Problem 2:

F_1 = Living forever by choosing box 1

F_2: Living forever by choosing box 2

M_1 = Living miserably by choosing box 1

M_2: Living miserably by choosing box 2

Box 1 : $(F_1 \wedge M_2)$

Box 2: $(F_1 \wedge M_2) \vee (F_2 \wedge M_1)$ $= (F_1 \wedge M_2) \oplus ((F_1 \wedge M_2) \vee (F_2 \wedge M_1))$

a. $((F_1 \wedge M_2) \wedge \neg((F_1 \wedge M_2) \vee (F_2 \wedge M_1))) \vee (\neg(F_1 \wedge M_2) \wedge ((F_1 \wedge M_2) \vee (F_2 \wedge M_1)))$

+ Box 1 is telling the truth — Box 2 is telling the truth

b. $((F_1 M_2 \cdot \overline{((F_1 M_2) + (F_2 M_1))}) + (\overline{(F_1 M_2} \ (F_1 M_2 + F_2 M_1))$

c. $F_1 M_2 \cdot \overline{(F_1 M_2 + F_2 M_1)} + (\overline{F_1 M_2}(F_1 M_2 + F_2 M_1))$

$= (F_1 M_2 \cdot \overline{F_1 M_2} \cdot \overline{F_2 M_1}) + (\overline{F_1 M_2} \ (F_1 M_2 + F_2 M_1)$

$= (0 \cdot \overline{F_2 M_1}) + (\overline{F_1 M_2} \ (F_1 M_2 + F_2 M_1)$

$= 0 + (\overline{F_1 M_2} \ (F_1 M_2 + F_2 M_1)$

$= \overline{F_1 M_2} \ (F_1 M_2 + F_2 M_1)$

$= \overline{F_1 M_2} \ F_1 M_2 + \overline{F_1 M_2} \ F_2 M_1$

$= 0 + \overline{F_1 M_2} \ F_2 M_1$

$= \overline{F_1 M_2} \cdot F_2 M_1$

d. You should choose box 2. If box one was chosen and F_1 = true then M_2 would also have to equate to true as the inscription says. This would then cause $F_1 M_2$ to be true be equal to true however the not of this is false and false/0 times anything would make the expression not true.

e. The inscription that is true is the one in box 2.

f. not mutually exclusive see truth table

Problem 2f

F_1	F_2	M_1	M_2	$F_1 \wedge M_2$	$F_2 \wedge M_1$	$\neg(F_1 \wedge M_2)$	$\neg(F_1 \wedge M_2) \wedge (F_2 \wedge M_1)$
0	0	0	0	0	0	1	0
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	0
0	1	0	0	0	0	1	0
0	1	0	1	0	0	1	0
0	1	1	0	0	1	1	1
0	1	1	1	0	1	1	1
1	0	0	0	0	0	1	0
1	0	0	1	1	0	0	0
1	0	1	0	0	0	1	0
1	0	1	1	1	0	0	0
1	1	0	0	0	0	1	0
1	1	0	1	1	0	0	0
1	1	1	0	0	1	1	1
1	1	1	1	1	1	0	0

where the expression is true

$\overline{F_1} F_2 M_1 \overline{M_2} \equiv T$

$\overline{F_1} F_2 M_1 M_2 \equiv T$

$F_1 F_2 M_1 \overline{M_2} \equiv T$

However by choosing F_2, F_1 would have to be false leaving:

1. $\overline{F_1} F_2 M_1 \overline{M_2}$
2. $\overline{F_1} F_2 M_1 M_2$

In scenario 1, the two are mutually exclusive

In scenario 2, they are not and you could live forever but also have a miserable life. meaning living forever and having a miserable life is not mutually exclusive

Problem 3:

a.

yes $x\bar{y}+zv$ is sansfiable

if $x=T, y=F, z=T, v=T$ the function $=T$ $(T\cdot T)+(T\cdot T)=T$

b. yes $(x+y)(\bar{x}+z)(\bar{y}+\bar{z})(x+v)$ is sansfiable

if $x=F, y=T, z=F, v=T$

$(F+\bar{T})(T+F)(F+T)(F+T) = T\cdot T\cdot T\cdot T = T$

c. NO, $x\bar{x}+y\bar{y}+z\bar{z}+v\bar{v}$ is not sansfiable

law: $p\bar{p}=F$ (due to complement laws. This law applies for every term in the equation and $F+F+F+F=($ therefore it is unsansfiable.

d. NO, $(x+y)(\overline{x+y+z})+(x+y+z)(\overline{x+y+z+v})$ is not sansfiable

once you apply the distribution law you get terms containing $x\bar{x}, y\bar{y}, \ldots$ which according to complement laws make $=F$ and when these terms are multiplied by others $p\cdot F=F$ due to identity law.

Problem 4:

a. $\bar{q}(\bar{p}+\bar{r})+\bar{p}(q+r)$

$=(\bar{q}\bar{p}+\bar{q}\bar{r})+(\bar{p}q)+\bar{p}r)$

$=\bar{q}\bar{p}+\bar{p}q+\bar{p}r+\bar{q}\bar{r}$

$=\bar{p}(\bar{q}+q+r)+\bar{q}\bar{r}$

$=\bar{p}(1+r)+\bar{q}\bar{r}$

$=\bar{p}(1)+\bar{q}\bar{r}$

$=\bar{p}+\bar{q}\bar{r}$

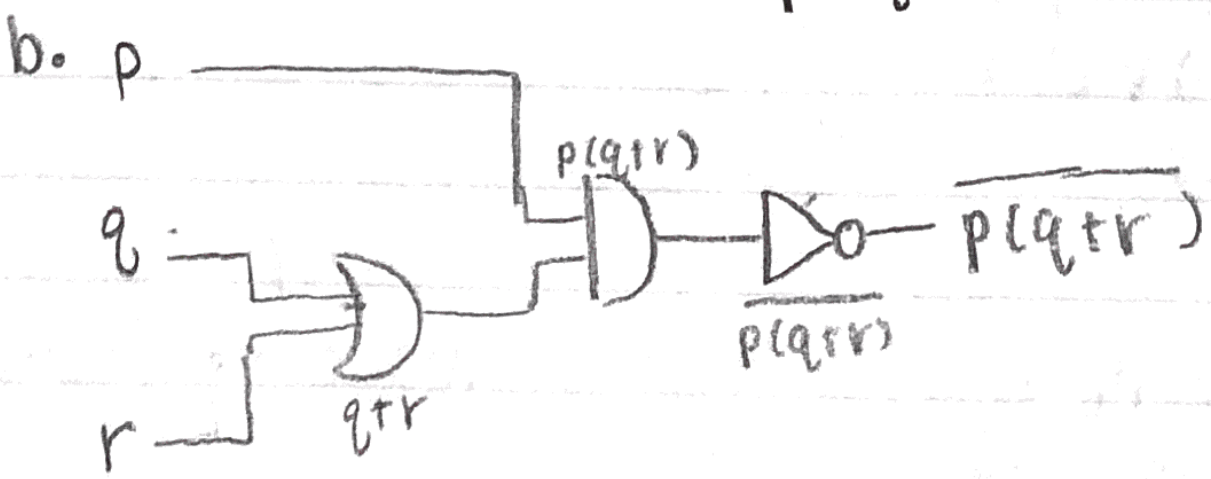
$=(\bar{p}+\bar{q})(\bar{p}+r)$

$=\overline{(pq)}\,\overline{(pr)}$

$=\overline{(pq)+(pr)}$

$=\overline{p(q+r)}$

b.



Problem 5:

The amount of functions possible is denoted by the formula: $2^{2^{\#\text{ of inputs}}}$, or, $2^{\#\text{ of rows}}$. This represents all possible functions. This formula gives the amount of binary operators as essentially the binary operators are functions on the inputs. The formula is derived from 2 options (0 or 1) ^ # of rows.

With two inputs p, q this means # of inputs = 2 and therefore number of rows = 4

there are $2^{2^2} = 16$ binary operators

Problem 6:

consider n=3:

$$\left[\bigvee_{i=1}^{2}\ \bigvee_{j=i+1}^{3}(p_i \wedge p_j)\right]$$

outer expansion:

$$\left[\bigvee_{j=2}^{3}(p_1 \wedge p_j)\right] \wedge \left[\bigvee_{j=3}^{3}(p_2 \wedge p_j)\right]$$

$$\downarrow$$

$$[(p_1 \wedge p_2) \vee (p_1 \wedge p_3)] \vee [(p_2 \wedge p_3)]$$

more than 1 true:

If $p_1 = T, p_2 = F, p_3 = T$

$(T \wedge F) \vee (T \wedge T) \vee (F \wedge T)$

$F \vee T \vee F$ Law of domination

$\equiv T$ makes all True

at most 1 is true:

If $p_1 = F, p_2 = T, p_3 = F$

$(F \wedge T) \vee (F \wedge F) \vee (T \wedge F)$

$F \vee F \vee F$

$\equiv F$

therefore if and only if at most there is one T value the expression is false

If more than one p_i value is true it makes the term $(p_{i_1} \wedge p_{i_2}) \equiv T$ and therefore the rest of the composition true according to the law of domination

Problem 7:

a. $(r_1 \wedge r_3) \vee (r_2 \wedge r_3) \vee (r_1 \wedge r_2)$

b. $(r2 \wedge r3) \equiv F$

c. $(r_1 \wedge r_3) \equiv F$

$(F) \vee (F) \vee (r_1 \wedge r_2) \equiv T$ identity law

$F \vee (r_1 \wedge r_2) \equiv T$ identity law

$r_1 \wedge r_2 \equiv T$ identity law

- r_1 is true
- r_2 is true
- r_3 is false

therefore chloes har is not req

$\neg r3$

when writing a class, you are not writing a program
- no static method

Accessor Methods
Access, don't change
public int getWidth(){ return width; }
returns value of variable
- width height private
dont want user to change
gives client access to + the data stored in private var.

}

mutator method
Alters the values of the data members
public void grow(int width, int height){
this.width = width;
this.height = height;

instance methods

setter methods:
responsible for preventing inappop. changes
public void setWidth(int w){
if (w <= 0){
throw new IllegalArgumentException()
}
this.width = w;
public void setHeight(int h){
if (h <= 0){
throw
}
this.height = h;
public void grow(int dw, int dh){
this.setWidth(width+dw)
this.setHeight(height+dh)
only one parameter but 2 things passed
}

mutator that scales dim. by a factor
public void scale(int factor) ← this
Accessor that determines if it is square
public boolean isSquare() ← passed address of object called on (this)