Exercise 1. Let the universe U = {1, 2, 3}. Expand the following expressions

into propositional term (i.e., remove the quantifiers):

(a) Vx. F(x)

(b) Ex. F(x)

(c) Ex. V y. G(x, y)

a)

F(1) /\ F(2) /\ F(3)

b)

F(1) \/ F(2) \/ F(3)

c)

(G(1,1)/\G(1,2) \/G(1,3)) \/

(G(2,1)/\G(2,2) \/G(2,3)) \/

(G(3,1)/\G(3,2) \/G(3,3))

**Exercise 2.** Let the universe be the set of integers. Expand the following

expression: *∀x ∈ {*1*,* 2*,* 3*,* 4*}. ∃y ∈ {*5*,* 6*}. F*(*x, y*)

(F(1,5) \/ F(1,6)) /\

(F(2,5) \/ F(2,6)) /\

(F(3,5) \/ F(3,6)) /\

(F(4,5) \/ F(4,6))

**Exercise 3.** Express the following statements formally, using the universe

of natural numbers, and the predicates *E*(*x*) *≡ x* is even and *O*(*x*) *≡*

*x* is odd.

*a)•* There is an even number.

*b)•* Every number is either even or odd.

*c)•* No number is both even and odd.

*d)•* The sum of two odd numbers is even.

*e)•* The sum of an odd number and an even number is odd.

a) ∃x . isEven(X)

*b) ∀x.isEven(x)\/ isOdd(X)*

*c) ∀x. ￢ (isEven(x)/\ isOdd(x))*

*d) ∀x∀y.isOdd(x) /\ isOdd(y) -> isEven(x+y)*

*∀x∀y.isOdd(x) /\ isEven(y) -> isOdd(x+y)*

**Exercise 4.** Let the universe be the set of all animals, and define the following

predicates:

*B*(*x*) *≡ x* is a bird.

*D*(*x*) *≡ x* is a dove.

*C*(*x*) *≡ x* is a chicken.

*P*(*x*) *≡ x* is a pig.

*F*(*x*) *≡ x* can fly.

*W*(*x*) *≡ x* has wings.

*M*(*x, y*) *≡ x* has more feathers than *y* does.

Translate the following sentences into logic. There are generally several

correct answers. Some of the English sentences are fairly close to logic,

while others require more interpretation before they can be rendered in

logic.

• Chickens are birds.

*∀x*. C(x) -> B(x)

• Some doves can fly.

*∃* x. D(x) /\ F(x)

• Pigs are not birds.

*∀x. P*(*x*) *-> ￢B*(*x*)

• Some birds can fly, and some can’t.

*∃x. B*(*x*) *∧ F*(*x*) *∧* *∃x. B*(*x*)*∧ ￢F*(*x*)

• An animal needs wings in order to fly.

*∀x.( ￢W(x)-> ￢F(x))*

• If a chicken can fly, then pigs have wings.

*∃x. C*(*x*) */\ F*(*x*) *-> ∀x. P*(*x*) *-> W*(*x*)

• Chickens have more feathers than pigs do.

*∀x. ∀y.* (*C*(*x*) */\ P*(*y*)) *-> M*(*x, y*)

• An animal with more feathers than any chicken can fly.

*∀x. A*(*x*) */\* (*∀y.* (*C*(*y*) */\ M*(*x, y*)))*-> F*(*x*)

**Exercise 5.** Translate the following into English.

*• ∀x.* (*∃ y.* wantsToDanceWith (*x, y*))

Para toda x que pertenece a chicos y, existe al menos una y que pertenece a chicas tal que quiere bailar con x.

*• ∃x.* (*∀ y.* wantsToPhone (*y, x*))

Todos tenemos alguien a quien queremos hablar por teléfono

*• ∃x.* (tired (*x*) *∧ ∀y.* helpsMoveHouse (*x, y*))

Alguien está cansado de ayudarle a los demás a mudarse de casa

**Exercise 6.** Write the predicate logic expressions corresponding to the following

Haskell expressions. Then decide whether the value is True or False,

and evaluate using the computer. Note that (== 2) is a function that

takes a number and compares it with 2, while (< 4) is a function that

takes a number and returns True if it is less than 4.

forall [1,2,3] (== 2)

forall [1,2,3] (< 4)

Like forall, the function exists applies its second argument to all of the

elements in its first argument:

exists :: [Int] -> (Int -> Bool) -> Bool

However, exists forms the disjunction of the result, using the Haskell function

or :: [Bool] -> Bool.

forall [1,2,3] (==2)

= 1==2 /\ 2==2 /\ 3==2

= False /\ True /\ False

= False

forall [1,2,3] (< 4)

= (1 < 4) /\ (2 < 4) /\ (3 < 4)

= True /\ True /\ True

= True

**Exercise 7.** Again, rewrite the following in predicate logic, work out the values

by hand and evaluate on the computer:

exists [0,1,2] (== 2)

exists [1,2,3] (> 5)

The functions exists and forall can be nested in the same way as quantifiers

can be nested in predicate logic. It’s convenient to express inner quantified

formulas as separate functions.

exists [0,1,2] (==2)

= 0==2 \/ 1==2 \/ 2==2

= False \/ False \/ True

= True

exists [1,2,3] (> 5)

= (1 > 5) \/ (2 > 5) \/ (3 > 5)

= False \/ False \/ False

= False

**Exercise 8.** Define the predicate *p x y* to mean *x* = *y*+1, and let the universe

be *{*1*,* 2*}*. Calculate the value of each of the following expressions, and

then check your solution using Haskell.

**(a)** *∀ x.* (*∃ y. p*(*x, y*))

**(b)** *∃ x, y. p*(*x, y*)

**(c)** *∃ x.* (*∀ y. p*(*x, y*))

**(d)** *∀ x, y. p*(*x, y*)

1. *∀x.*(*∃y.p*(*x, y*))

= *∀x. x* = 1+1 *∨ x* = 2+1

= (1 = 1+1 *∨* 1 = 2+1) *∧* (2 = 1 + 1 *∨* 2 = 2+1)

= (False *∨* False) *∧* (True *∨* False)

= False *∧* True

= False

2. *∃x.*(*∃y.p*(*x, y*))

= *∃x.*(*x* = 1+1) *∨* (*x* = 2+1)

= (1 = 1+1 *∨* 1 = 2+1) *∨* (2 = 1 + 1 *∨* 2 = 2+1)

= (False *∨* False) *∨* (True *∨* False)

= False *∨* True

= True

3. *∃x.*(*∀y.p*(*x, y*))

= *∃x.*(*x* = 1+1 *∧ x* = 2+1)

= (1 = 1+1 *∧* 1 = 2+1) *∨* (2 = 1 + 1 *∧* 2 = 2+1)

= (False *∧* False) *∨* (True *∧* False)

= False *∨* False

= False

4. *∀x, y.p*(*x, y*)

= *∀x.*(*x* = 1+1 *∧ x* = 2+1)

= (1 = 1 + 1 *∧* 1 = 2+1) *∧* (2 = 1 + 1 *∧* 2 = 2+1)

= (False *∧* False) *∧* (True *∧* False)

= False *∧* False

= False

**Exercise 9.** Prove *∀x.F* (*x*)*, ∀x.F* (*x*) *→ G*(*x*) *\_ ∀x.G*(*x*).

*∀x.F* (*x*)*, ∀x.F* (*x*) *→ G*(*x*) *\_ ∀x.G*(*x*)

*Proof.*

*∀x.F* (*x*) *∀x.F* (*x*) *→ G*(*x*)

*- - - - - - - - -{∀E} - - - - - - - - - - - - - - - - -{∀E}*

*F*(*p*) *F*(*p*) *→ G*(*p*)

*- - - - - - - - - - - - - - - -- - - - - - -- - - - - -- - - - - -- - -{→E}*

*G*(*p*)

----------- *{∀I}*

*∀x.G*(*x*)

**Exercise 10.** Prove *∃x. ∃y. F*(*x, y*) *|- ∃y. ∃x. F*(*x, y*).

*∃x.∃y.F*(*x, y*) |- *∃y.∃x.F* (*x, y*)

*F*(*p, q*)

----------*{∃I}*

*∃y.F*(*p, y*) *∃x.F* (*x, q*)

*- - - - - - - -- - - - - - -- - - - - - - -{∃E}*

*∃x.F* (*x, q*)

*- - - - - -- - -- - - - -- - - -- - - - - - -- {∃I}*

*∃y.∃x.F* (*x, y*) *∃y.∃x.F* (*x, y*)

*- - - - - - - -- - - -- - - - -- - - - - - - -- - - - - - - - - - - - - -- - - - - - - -{∃E}*

*∃x.∃y.F*(*x, y*)

**Exercise 11.** The converse of Theorem 66 is the following:

*∀y. ∃x. F*(*x, y*) *|- ∃x. ∀y. F*(*x, y*) ***Wrong!***

Give a counterexample that demonstrates that this statement is *not* valid.

*∀y.∃x.F* (*x, y*) *|- ∃x.∀y.F*(*x, y*)

y=x+y

X=[0,1,2]

Y=[0,1,2]

(F(0,0) \/ F(0,1) \/ F(0,2)) /\

(F(1,0) \/ F(1,1) \/ F(1,2)) /\

(F(2,0) \/ F(2,1) \/ F(2,2))

(T /\ T /\ T) (T /\ T /\ T)

/\ (F /\ F /\ F) /\ (F /\ F /\ F)

/\ (F /\ F /\ F) /\ (F /\ F /\ F)

----------------------- ---------------------

FALSE =/= TRUE

WRONG!

**Exercise 12.** Prove *∀x.*(*F*(*x*) *∧ G*(*x*)) *|-* (*∀x.F* (*x*)) *∧* (*∀x.G*(*x*)).

*Proof.*

*∀x.F* (*x*) *∧ G*(*x*) *∀x.F* (*x*) *∧ G*(*x*)

--- - - - - - - - - - -*{∀E} - - - - - - - - - - - - - - - - - - {∀E}*

*F*(*p*) *∧ G*(*p*) *F*(*q*) *∧ G*(*q*)

*- - - - - - - - - - - - - - - -{∧EL} - - - - - - - - - - - - - - - - - {∧ER}*

*F*(*p*) *G*(*q*)

*- - - - - - - - - - {∀I} - - - - - - - - - - - - - -{∀I}*

*∀x.F* (*x*) *∀x.G*(*x*)

*- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -- - - {∧I}*

*∀x.F* (*x*) *∧ ∀x.G*(*x*)