Computación y Estructuras Discretas I

Andrés A. Aristizábal P.
aaaristizabal@icesi.edu.co

Departamento de Tecnologías de Información y Comunicaciones

UNIVERSIDAD ICESI

2024-2

Agenda del día

1. **Métodos de demostración**
   - Demostraciones en Coq

2. **Fundamentos de Conjuntos**
   - Presentación del siguiente tema

Agenda del día

1. **Métodos de demostración**
   - Demostraciones en Coq
2. **Fundamentos de Conjuntos**
   - Presentación del siguiente tema

¿Qué es una declaración en Coq?

¿Qué es una declaración en Coq?

Una declaración asocia un nombre con una especificación.

¿Qué es una declaración en Coq?

Una declaración asocia un nombre con una especificación.

¿Qué es una especificación en Coq?

¿Qué es una declaración en Coq?

Una declaración asocia un nombre con una especificación.

¿Qué es una especificación en Coq?

Es una expresión formal que clasifica la noción que está declarando.

¿Qué es una declaración en Coq?

Una declaración asocia un nombre con una especificación.

¿Qué es una especificación en Coq?

Es una expresión formal que clasifica la noción que está declarando.

- Hay tres clases de especificaciones: proposiciones lógicas, colecciones matemáticas y tipos abrstractos.

¿Qué es una declaración en Coq?

Una declaración asocia un nombre con una especificación.

¿Qué es una especificación en Coq?

Es una expresión formal que clasifica la noción que está declarando.

- Hay tres clases de especificaciones: proposiciones lógicas, colecciones matemáticas y tipos abrstractos.
- Estas se clasifican en tres tipos: *Prop, Set* y *Type.*

¿Cuáles son algunas definiciones?

¿Cuáles son algunas definiciones?

- `nat` es una definición aritmética, una colección matemática.

¿Cuáles son algunas definiciones?

- `nat` es una definición aritmética, una colección matemática.
- Las constantes `O`, `S` y `plus` se definen como objetos de tipo `nat`, `nat -> nat`, y `nat -> nat -> nat`.

¿Cuáles son algunas definiciones?

- `nat` es una definición aritmética, una colección matemática.
- Las constantes `O`, `S` y `plus` se definen como objetos de tipo `nat`, `nat -> nat`, y `nat -> nat -> nat`.
- También se pueden introducir nuevas definiciones con un nombre y un tipo bien definido

Ejemplo

Definición de uno

```
Coq < Definition one := (S O).
one is defined
```

Ejemplo

Definición de dos

Ejemplo

Definición de dos

```
Coq < Definition two : nat := S one.
two is defined
```

Ejemplo

Definición de doblar un número

Ejemplo

Definición de doblar un número

```
Coq < Definition double (m : nat) := plus m m.
double is defined
```

Introducción al motor de demostraciones

Introducción al motor de demostraciones

- Vamos a verificar la veracidad de una tautología.

Introducción al motor de demostraciones

- Vamos a verificar la veracidad de una tautología.
- $((A \rightarrow (B \rightarrow C)) \rightarrow (A \rightarrow B) \rightarrow (A \rightarrow C)$

Introducción al motor de demostraciones

- Vamos a verificar la veracidad de una tautología.
- $((A \rightarrow (B \rightarrow C)) \rightarrow (A \rightarrow B) \rightarrow (A \rightarrow C)$
- Definimos el teorema o lema a verificar.

```
Theorem theo : forall A B C : Prop, (A -> B -> C) -> (A -> B) -> A -> C.
```

Demostraciones en Coq

```
1 goal
______________________________________
forall A B C : Prop, (A -> B -> C) -> (A -> B) -> A -> C
```

¿Qué ocurre en este momento?

¿Qué ocurre en este momento?

- El sistema muestra la meta u objetivo debajo de la línea (lo que queremos demostrar).

¿Qué ocurre en este momento?

- El sistema muestra la meta u objetivo debajo de la línea (lo que queremos demostrar).
- Las hipótesis locales se presentan arriba de dicha línea (en este caso aún no hay).

¿Qué ocurre en este momento?

- El sistema muestra la meta u objetivo debajo de la línea (lo que queremos demostrar).
- Las hipótesis locales se presentan arriba de dicha línea (en este caso aún no hay).
- Esta combinación de hipótesis locales y metas se denomina juicio.

¿Qué ocurre en este momento?

- El sistema muestra la meta u objetivo debajo de la línea (lo que queremos demostrar).
- Las hipótesis locales se presentan arriba de dicha línea (en este caso aún no hay).
- Esta combinación de hipótesis locales y metas se denomina juicio.
- En este momento nos encontramos en un ciclo interno del sistema que se denomina modo de demostración.

¿Qué ocurre en este momento?

- El sistema muestra la meta u objetivo debajo de la línea (lo que queremos demostrar).
- Las hipótesis locales se presentan arriba de dicha línea (en este caso aún no hay).
- Esta combinación de hipótesis locales y metas se denomina juicio.
- En este momento nos encontramos en un ciclo interno del sistema que se denomina modo de demostración.
- En este modo existen comandos denominados tácticas que nos ayudaran a alcanzar nuestra meta.

¿Qué son las tácticas?

¿Qué son las tácticas?

- Estrategias de demostración.

¿Qué son las tácticas?

- Estrategias de demostración.
- Operan en la meta actual buscando construir una prueba para el juicio correspondiente.

¿Qué son las tácticas?

- Estrategias de demostración.
- Operan en la meta actual buscando construir una prueba para el juicio correspondiente.
- Pueden partir de pruebas de juicios hipotéticos.

¿Qué son las tácticas?

- Estrategias de demostración.
- Operan en la meta actual buscando construir una prueba para el juicio correspondiente.
- Pueden partir de pruebas de juicios hipotéticos.
- Pueden generar conjeturas que se añaden a la lista de elementos útiles para resolver el juicio original.

¿Qué son las tácticas?

- Estrategias de demostración.
- Operan en la meta actual buscando construir una prueba para el juicio correspondiente.
- Pueden partir de pruebas de juicios hipotéticos.
- Pueden generar conjeturas que se añaden a la lista de elementos útiles para resolver el juicio original.
- Por ejemplo, la táctica `intro` se puede aplicar a cualquier juicio en el cual su meta es una implicación.

¿Qué son las tácticas?

- Estrategias de demostración.
- Operan en la meta actual buscando construir una prueba para el juicio correspondiente.
- Pueden partir de pruebas de juicios hipotéticos.
- Pueden generar conjeturas que se añaden a la lista de elementos útiles para resolver el juicio original.
- Por ejemplo, la táctica `intro` se puede aplicar a cualquier juicio en el cual su meta es una implicación.
- Mueve la proposición a la izquierda de la aplicación a la lista de hipótesis locales

Introducción al motor de demostraciones

Introducción al motor de demostraciones

- Aplicamos la táctica intros que permite aplicar la táctica intro múltiples veces.

Introducción al motor de demostraciones

- Aplicamos la táctica intros que permite aplicar la táctica intro múltiples veces.

```
intros.
```

```
A,B,C : Prop
H: A -> B -> C
H0: A -> B
H1: A
____________
C
```

Introducción al motor de demostraciones

Introducción al motor de demostraciones

- Nos damos cuenta que la meta es C

Introducción al motor de demostraciones

- Nos damos cuenta que la meta es C
- Para obtenerla simplemente aplicamos la hipótesis H.

Introducción al motor de demostraciones

- Nos damos cuenta que la meta es C
- Para obtenerla simplemente aplicamos la hipótesis H.

```
apply H.
2 subgoals

  A, B, C : Prop
  H : A -> B -> C
  H0 : A -> B
  H1 : A
______________
  A
subgoal 2 is:
 B
```

Introducción al motor de demostraciones

Introducción al motor de demostraciones

- Tenemos dos submetas A y B

Introducción al motor de demostraciones

- Tenemos dos submetas A y B
- La primera ya la tenemos con la hipótesis H1

Introducción al motor de demostraciones

- Tenemos dos submetas A y B
- La primera ya la tenemos con la hipótesis H1
- Por tal aplicamos la táctica `exact H1`

Introducción al motor de demostraciones

- Tenemos dos submetas A y B
- La primera ya la tenemos con la hipótesis H1
- Por tal aplicamos la táctica `exact H1`

```
exact H1.
1 subgoal

  A, B, C : Prop
  H : A -> B -> C
  H0 : A -> B
  H1 : A
______________
  B
```

Introducción al motor de demostraciones

Introducción al motor de demostraciones

- Si aplicamos $H1$ ya necesitamos solamente A

Introducción al motor de demostraciones

- Si aplicamos $H1$ ya necesitamos solamente A

```
apply H0.
1 subgoal

  A, B, C : Prop
  H : A -> B -> C
  H0 : A -> B
  H1 : A
______________
  A
```

Introducción al motor de demostraciones

Introducción al motor de demostraciones

- Utilizamos `exact H1` o simplemente `assumption` (que se puede resolver a partir de las premisas previas)

Introducción al motor de demostraciones

- Utilizamos `exact H1` o simplemente `assumption` (que se puede resolver a partir de las premisas previas)
- Para verificar y terminar se usa `Qed`

¿Cuáles son algunas de estas tácticas útiles para resolver metas sencillas?

¿Cuáles son algunas de estas tácticas útiles para resolver metas sencillas?

- `assumption`: resuelve una meta que ya se supone en el contexto

¿Cuáles son algunas de estas tácticas útiles para resolver metas sencillas?

- `assumption`: resuelve una meta que ya se supone en el contexto
- `reflexivity`: resuelve una meta si existe una igualdad trivial

¿Cuáles son algunas de estas tácticas útiles para resolver metas sencillas?

- `assumption`: resuelve una meta que ya se supone en el contexto
- `reflexivity`: resuelve una meta si existe una igualdad trivial
- `discriminate`: resuelve una meta si es una desigualdad trivial o una falsa igualdad

¿Cuáles son algunas de estas tácticas útiles para resolver metas sencillas?

- `assumption`: resuelve una meta que ya se supone en el contexto
- `reflexivity`: resuelve una meta si existe una igualdad trivial
- `discriminate`: resuelve una meta si es una desigualdad trivial o una falsa igualdad
- `exact`: resuelve una meta si se conoce previamente el término que demuestra la meta

¿Cuáles son algunas de estas tácticas útiles para resolver metas sencillas?

- `assumption`: resuelve una meta que ya se supone en el contexto
- `reflexivity`: resuelve una meta si existe una igualdad trivial
- `discriminate`: resuelve una meta si es una desigualdad trivial o una falsa igualdad
- `exact`: resuelve una meta si se conoce previamente el término que demuestra la meta
- `contradiction`: resuelve una meta si el contexto contiene `False` o una hipótesis contradictoria

¿Cuáles son algunas tácticas útiles para transformar metas?

¿Cuáles son algunas tácticas útiles para transformar metas?

- `intro / intros`: introducen variables definidas o términos que aparecen a la izquierda de las implicaciones

¿Cuáles son algunas tácticas útiles para transformar metas?

- `intro / intros`: introducen variables definidas o términos que aparecen a la izquierda de las implicaciones
- `simpl`: simplifica una meta o una hipótesis dentro del contexto

¿Cuáles son algunas tácticas útiles para transformar metas?

- `intro / intros`: introducen variables definidas o términos que aparecen a la izquierda de las implicaciones
- `simpl`: simplifica una meta o una hipótesis dentro del contexto
- `apply`: utiliza implicaciones para transformar metas e hipótesis

¿Cuáles son algunas tácticas útiles para transformar metas?

- `intro / intros`: introducen variables definidas o términos que aparecen a la izquierda de las implicaciones
- `simpl`: simplifica una meta o una hipótesis dentro del contexto
- `apply`: utiliza implicaciones para transformar metas e hipótesis
- `rewrite`: reemplaza un término con otro equivalente si dicha equivalencia ya ha sido demostrada

Demostraciones en Coq

¿Cuáles son algunas tácticas útiles para transformar metas?

- `intro / intros`: introducen variables definidas o términos que aparecen a la izquierda de las implicaciones
- `simpl`: simplifica una meta o una hipótesis dentro del contexto
- `apply`: utiliza implicaciones para transformar metas e hipótesis
- `rewrite`: reemplaza un término con otro equivalente si dicha equivalencia ya ha sido demostrada
- `left / right`: reemplaza una meta que consiste de una disyunción $P \vee Q$ con solamente P o Q

¿Cuáles son algunas tácticas útiles para dividir metas e hipótesis?

¿Cuáles son algunas tácticas útiles para dividir metas e hipótesis?

- `split`: reemplaza una meta que consiste de una conjunción P / \Q con dos submetas P y Q

¿Cuáles son algunas tácticas útiles para dividir metas e hipótesis?

- `split`: reemplaza una meta que consiste de una conjunción P / \Q con dos submetas P y Q
- `destruct (and/or)`: reemplaza una hipótesis P / \Q con dos hipótesis P y Q. De manera alternativa si la hipótesis es una disyunción P \/ Q, genera dos submetas, una donde P se mantiene y otra donde Q se mantiene.

¿Cuáles son algunas tácticas útiles para dividir metas e hipótesis?

- `split`: reemplaza una meta que consiste de una conjunción P / \Q con dos submetas P y Q
- `destruct (and/or)`: reemplaza una hipótesis P / \Q con dos hipótesis P y Q. De manera alternativa si la hipótesis es una disyunción P \/ Q, genera dos submetas, una donde P se mantiene y otra donde Q se mantiene.
- `destruct (análisis por casos)`: genera una submeta para cada constructor de tipo inductivo

¿Cuáles son algunas tácticas útiles para dividir metas e hipótesis?

- `split`: reemplaza una meta que consiste de una conjunción P / \Q con dos submetas P y Q
- `destruct (and/or)`: reemplaza una hipótesis P / \Q con dos hipótesis P y Q. De manera alternativa si la hipótesis es una disyunción P \/ Q, genera dos submetas, una donde P se mantiene y otra donde Q se mantiene.
- `destruct (análisis por casos)`: genera una submeta para cada constructor de tipo inductivo
- `induction`: genera una submeta para cada constructor de tipo inductivo y provee una hipótesis inductiva para constructores definidos recursivamente

¿Cómo realizar una demostración?

¿Cómo realizar una demostración?

$P \rightarrow P$

Demostraciones en Coq

¿Cómo realizar una demostración?

$P \rightarrow P$

```
Lemma id_P : forall P : Prop, P -> P.
```

¿Cómo realizar una demostración?

$P \to P$

```
Lemma id_P : forall P : Prop, P -> P.
intro P.
```

¿Cómo realizar una demostración?

$P \rightarrow P$

```
Lemma id_P : forall P : Prop, P -> P.
intro P.
intro p.
```

Demostraciones en Coq

¿Cómo realizar una demostración?

$P \rightarrow P$

```
Lemma id_P : forall P : Prop, P -> P.
intro P.
intro p.
exact p.
```

¿Cómo realizar una demostración?

$P \to P$

```
Lemma id_P : forall P : Prop, P -> P.
intro P.
intro p.
exact p.
Qed.
```

Ejemplo

Ejemplo

$(P \rightarrow Q) \rightarrow (Q \rightarrow R) \rightarrow P \rightarrow R$

Ejemplo

$(P \to Q) \to (Q \to R) \to P \to R$

```
Lemma imp_trans : forall P Q R : Prop,
(P->Q)->(Q->R)->P->R.
```

Ejemplo

$(P \to Q) \to (Q \to R) \to P \to R$

```
Lemma imp_trans : forall P Q R : Prop,
(P->Q)->(Q->R)->P->R.
intros P Q R.
```

Ejemplo

$(P \to Q) \to (Q \to R) \to P \to R$

```
Lemma imp_trans : forall P Q R : Prop,
(P->Q)->(Q->R)->P->R.
intros P Q R.
intros firstimp secimp p.
```

Ejemplo

$(P \to Q) \to (Q \to R) \to P \to R$

```
Lemma imp_trans : forall P Q R : Prop,
(P->Q)->(Q->R)->P->R.
intros P Q R.
intros firstimp secimp p.
apply secimp.
```

Ejemplo

$(P \to Q) \to (Q \to R) \to P \to R$

```
Lemma imp_trans : forall P Q R : Prop,
(P->Q)->(Q->R)->P->R.
intros P Q R.
intros firstimp secimp p.
apply secimp.
apply firstimp.
```

Ejemplo

$(P \to Q) \to (Q \to R) \to P \to R$

```
Lemma imp_trans : forall P Q R : Prop,
(P->Q)->(Q->R)->P->R.
intros P Q R.
intros firstimp secimp p.
apply secimp.
apply firstimp.
exact p.
```

Ejemplo

$(P \to Q) \to (Q \to R) \to P \to R$

```
Lemma imp_trans : forall P Q R : Prop,
(P->Q)->(Q->R)->P->R.
intros P Q R.
intros firstimp secimp p.
apply secimp.
apply firstimp.
exact p.
Qed.
```

Ejemplo

Ejemplo

Demuestre que las premisas “Juliana es una gran matemática” y “Martina es una excelente nadadora” permiten concluir que “Martina es una excelente nadadora y Juliana es una gran matemática”.

Ejemplo

Demuestre que las premisas “Juliana es una gran matemática” y “Martina es una excelente nadadora” permiten concluir que “Martina es una excelente nadadora y Juliana es una gran matemática”.

P : “Juliana es una gran matemática”

Q : “Martina es una excelente nadadora”

Ejemplo

Demuestre que las premisas “Juliana es una gran matemática” y “Martina es una excelente nadadora” permiten concluir que “Martina es una excelente nadadora y Juliana es una gran matemática”.

P : “Juliana es una gran matemática”

Q : “Martina es una excelente nadadora”

```
Lemma and_comm : forall P Q : Prop, (P/ \Q)->(Q /
\P).
```

Ejemplo

Demuestre que las premisas "Juliana es una gran matemática" y "Martina es una excelente nadadora" permiten concluir que "Martina es una excelente nadadora y Juliana es una gran matemática".

P : "Juliana es una gran matemática"

Q : "Martina es una excelente nadadora"

```
Lemma and_comm : forall P Q : Prop, (P/ \Q)->(Q /
\P).
intros P Q.
```

Ejemplo

Demuestre que las premisas “Juliana es una gran matemática” y “Martina es una excelente nadadora” permiten concluir que “Martina es una excelente nadadora y Juliana es una gran matemática”.

P : “Juliana es una gran matemática”

Q : “Martina es una excelente nadadora”

```
Lemma and_comm : forall P Q : Prop, (P/ \Q)->(Q /
\P).
intros P Q.
intro pandq.
```

Ejemplo

Demuestre que las premisas “Juliana es una gran matemática” y “Martina es una excelente nadadora” permiten concluir que “Martina es una excelente nadadora y Juliana es una gran matemática”.

P : “Juliana es una gran matemática”

Q : “Martina es una excelente nadadora”

```
Lemma and_comm : forall P Q : Prop, (P/ \Q)->(Q /
\P).
intros P Q.
intro pandq.
destruct pandq as [p q].
```

Ejemplo

Demuestre que las premisas “Juliana es una gran matemática” y “Martina es una excelente nadadora” permiten concluir que “Martina es una excelente nadadora y Juliana es una gran matemática”.
P : “Juliana es una gran matemática”
Q : “Martina es una excelente nadadora”

```
Lemma and_comm : forall P Q : Prop, (P/ \Q)->(Q /
\P).
intros P Q.
intro pandq.
destruct pandq as [p q].
split.
```

Ejemplo

Demuestre que las premisas “Juliana es una gran matemática” y “Martina es una excelente nadadora” permiten concluir que “Martina es una excelente nadadora y Juliana es una gran matemática”.
P : “Juliana es una gran matemática”
Q : “Martina es una excelente nadadora”

```
Lemma and_comm : forall P Q : Prop, (P/ \Q)->(Q /
\P).
intros P Q.
intro pandq.
destruct pandq as [p q].
split.
exact q.
```

Demostraciones en Coq

Ejemplo

Demuestre que las premisas “Juliana es una gran matemática” y “Martina es una excelente nadadora” permiten concluir que “Martina es una excelente nadadora y Juliana es una gran matemática”.

P : “Juliana es una gran matemática”

Q : “Martina es una excelente nadadora”

```
Lemma and_comm : forall P Q : Prop, (P/ \Q)->(Q /
\P).
intros P Q.
intro pandq.
destruct pandq as [p q].
split.
exact q.
exact p.
```

Ejemplo

Demuestre que las premisas “Juliana es una gran matemática” y “Martina es una excelente nadadora” permiten concluir que “Martina es una excelente nadadora y Juliana es una gran matemática”.
P : “Juliana es una gran matemática”
Q : “Martina es una excelente nadadora”

```
Lemma and_comm : forall P Q : Prop, (P/ \Q)->(Q /
\P).
intros P Q.
intro pandq.
destruct pandq as [p q].
split.
exact q.
exact p.
Qed.
```

Ejemplo

Ejemplo

Demuestre que la premisa “Carlos va a salir de compras o se va a quedar viendo televisión” permite concluir que “Carlos se va a quedar viendo televisión o va a salir de compras”.

Ejemplo

Demuestre que la premisa “Carlos va a salir de compras o se va a quedar viendo televisión” permite concluir que “Carlos se va a quedar viendo televisión o va a salir de compras”.
P : “Carlos va a salir de compras”
Q : “Carlos se va a quedar viendo televisión”

Ejemplo

Demuestre que la premisa “Carlos va a salir de compras o se va a quedar viendo televisión” permite concluir que “Carlos se va a quedar viendo televisión o va a salir de compras”.
P : “Carlos va a salir de compras”
Q : “Carlos se va a quedar viendo televisión”

```
Lemma or_comm : forall P Q : Prop, (P \/ Q)->(Q \/
P).
```

Ejemplo

Demuestre que la premisa “Carlos va a salir de compras o se va a quedar viendo televisión” permite concluir que “Carlos se va a quedar viendo televisión o va a salir de compras”.

P : “Carlos va a salir de compras”

Q : “Carlos se va a quedar viendo televisión”

```
Lemma or_comm : forall P Q : Prop, (P \/ Q)->(Q \/
P).
intros P Q.
```

Ejemplo

Demuestre que la premisa “Carlos va a salir de compras o se va a quedar viendo televisión” permite concluir que “Carlos se va a quedar viendo televisión o va a salir de compras”.

P : “Carlos va a salir de compras”

Q : “Carlos se va a quedar viendo televisión”

```
Lemma or_comm : forall P Q : Prop, (P \/ Q)->(Q \/
P).
intros P Q.
intro porq.
```

Ejemplo

Demuestre que la premisa “Carlos va a salir de compras o se va a quedar viendo televisión” permite concluir que “Carlos se va a quedar viendo televisión o va a salir de compras”.
P : “Carlos va a salir de compras”
Q : “Carlos se va a quedar viendo televisión”

```
Lemma or_comm : forall P Q : Prop, (P \/ Q)->(Q \/
P).
intros P Q.
intro porq.
destruct porq as [p | q].
```

Ejemplo

Demuestre que la premisa “Carlos va a salir de compras o se va a quedar viendo televisión” permite concluir que “Carlos se va a quedar viendo televisión o va a salir de compras”.

P : “Carlos va a salir de compras”

Q : “Carlos se va a quedar viendo televisión”

```
Lemma or_comm : forall P Q : Prop, (P \/ Q)->(Q \/
P).
intros P Q.
intro porq.
destruct porq as [p | q].
right.
```

Ejemplo

Demuestre que la premisa “Carlos va a salir de compras o se va a quedar viendo televisión” permite concluir que “Carlos se va a quedar viendo televisión o va a salir de compras”.
P : “Carlos va a salir de compras”
Q : “Carlos se va a quedar viendo televisión”

```
Lemma or_comm : forall P Q : Prop, (P \/ Q)->(Q \/
P).
intros P Q.
intro porq.
destruct porq as [p | q].
right.
exact p.
```

Ejemplo

Demuestre que la premisa “Carlos va a salir de compras o se va a quedar viendo televisión” permite concluir que “Carlos se va a quedar viendo televisión o va a salir de compras”.

P : “Carlos va a salir de compras”

Q : “Carlos se va a quedar viendo televisión”

```
Lemma or_comm : forall P Q : Prop, (P \/ Q)->(Q \/
P).
intros P Q.
intro porq.
destruct porq as [p | q].
right.
exact p.
left.
```

Ejemplo

Demuestre que la premisa "Carlos va a salir de compras o se va a quedar viendo televisión" permite concluir que "Carlos se va a quedar viendo televisión o va a salir de compras".
P : "Carlos va a salir de compras"
Q : "Carlos se va a quedar viendo televisión"

```
Lemma or_comm : forall P Q : Prop, (P \/ Q)->(Q \/
P).
intros P Q.
intro porq.
destruct porq as [p | q].
right.
exact p.
left.
exact q.
```

Ejemplo

Demuestre que la premisa “Carlos va a salir de compras o se va a quedar viendo televisión” permite concluir que “Carlos se va a quedar viendo televisión o va a salir de compras”.

P : “Carlos va a salir de compras”

Q : “Carlos se va a quedar viendo televisión”

```
Lemma or_comm : forall P Q : Prop, (P \/ Q)->(Q \/
P).
intros P Q.
intro porq.
destruct porq as [p | q].
right.
exact p.
left.
exact q.
Qed.
```

Ejemplo

Ejemplo

Demuestre que las premisas “A Andrés le gusta jugar tenis” y “A Andrés le gusta jugar fútbol” permiten concluir que “A Andrés le gusta jugar tenis o fútbol”.

Ejemplo

Demuestre que las premisas “A Andrés le gusta jugar tenis” y “A Andrés le gusta jugar fútbol” permiten concluir que “A Andrés le gusta jugar tenis o fútbol”.

P : “A Andrés le gusta jugar tenis”
Q : “A Andrés le gusta jugar fútbol”

Demostraciones en Coq

Ejemplo

Demuestre que las premisas “A Andrés le gusta jugar tenis” y “A Andrés le gusta jugar fútbol” permiten concluir que “A Andrés le gusta jugar tenis o fútbol”.
P : “A Andrés le gusta jugar tenis”
Q : “A Andrés le gusta jugar fútbol”

```
Lemma example1 : forall P Q : Prop, (P/ \Q)->(P\/Q).
```

Ejemplo

Demuestre que las premisas “A Andrés le gusta jugar tenis” y “A Andrés le gusta jugar fútbol” permiten concluir que “A Andrés le gusta jugar tenis o fútbol”.
P : “A Andrés le gusta jugar tenis”
Q : “A Andrés le gusta jugar fútbol”

```
Lemma example1 : forall P Q : Prop, (P/ \Q)->(P\/Q).
intros P Q.
```

Ejemplo

Demuestre que las premisas “A Andrés le gusta jugar tenis” y “A Andrés le gusta jugar fútbol” permiten concluir que “A Andrés le gusta jugar tenis o fútbol”.

P : “A Andrés le gusta jugar tenis”

Q : “A Andrés le gusta jugar fútbol”

```
Lemma example1 : forall P Q : Prop, (P/ \Q)->(P\/Q).
intros P Q.
intro pandq.
```

Ejemplo

Demuestre que las premisas "A Andrés le gusta jugar tenis" y "A Andrés le gusta jugar fútbol" permiten concluir que "A Andrés le gusta jugar tenis o fútbol".

P : "A Andrés le gusta jugar tenis"

Q : "A Andrés le gusta jugar fútbol"

```
Lemma example1 : forall P Q : Prop, (P/ \Q)->(P\/Q).
intros P Q.
intro pandq.
destruct pandq as [p q].
```

Ejemplo

Demuestre que las premisas “A Andrés le gusta jugar tenis” y “A Andrés le gusta jugar fútbol” permiten concluir que “A Andrés le gusta jugar tenis o fútbol”.

P : “A Andrés le gusta jugar tenis”

Q : “A Andrés le gusta jugar fútbol”

```
Lemma example1 : forall P Q : Prop, (P/ \Q)->(P\/Q).
intros P Q.
intro pandq.
destruct pandq as [p q].
right.
```

Demostraciones en Coq

Ejemplo

Demuestre que las premisas “A Andrés le gusta jugar tenis” y “A Andrés le gusta jugar fútbol” permiten concluir que “A Andrés le gusta jugar tenis o fútbol”.

P : “A Andrés le gusta jugar tenis”

Q : “A Andrés le gusta jugar fútbol”

```
Lemma example1 : forall P Q : Prop, (P/ \Q)->(P\/Q).
intros P Q.
intro pandq.
destruct pandq as [p q].
right.
exact q.
```

Ejemplo

Demuestre que las premisas “A Andrés le gusta jugar tenis” y “A Andrés le gusta jugar fútbol” permiten concluir que “A Andrés le gusta jugar tenis o fútbol”.

P : “A Andrés le gusta jugar tenis”

Q : “A Andrés le gusta jugar fútbol”

```
Lemma example1 : forall P Q : Prop, (P/ \Q)->(P\/Q).
intros P Q.
intro pandq.
destruct pandq as [p q].
right.
exact q.
Qed.
```

Agenda del día

1. **Métodos de demostración**
   - Demostraciones en Coq

2. **Fundamentos de Conjuntos**
   - Presentación del siguiente tema

Preguntas de interés

- ¿Qué es un conjunto?

Preguntas de interés

- ¿Qué es un conjunto?

Definición

Es una colección desordenada de objetos.

Preguntas de interés

- ¿Qué es un conjunto?

Definición

Es una colección desordenada de objetos.

Ejemplo

$V = \{a, e, i, o, u\}$, *el conjuntos de las vocales*

Preguntas de interés

- ¿Qué es un conjunto?

Definición

Es una colección desordenada de objetos.

Ejemplo

$V = \{a, e, i, o, u\}$, *el conjuntos de las vocales*
$\mathbb{N} = \{0, 1, 2, 3, ...\}$, *el conjunto de los números naturales*

- ¿Qué es un conjunto?

Definición

Es una colección desordenada de objetos.

Ejemplo

$V = \{a, e, i, o, u\}$, *el conjuntos de las vocales*
$\mathbb{N} = \{0, 1, 2, 3, ...\}$, *el conjunto de los números naturales*
$\mathbb{Z} = \{..., -3, -2, -1, 0, 1, 2, 3, ...\}$, *el conjunto de los números enteros*

Preguntas de interés

- ¿Qué es un elemento de un conjunto?

Preguntas de interés

- ¿Qué es un elemento de un conjunto?

Definición

Los objetos de un conjunto se llaman también elementos o miembros del conjunto. Se dice que un conjunto contiene a sus elementos. Cuando objeto es un elemento de un conjunto, se dice que pertenece a dicho conjunto.

Preguntas de interés

- ¿Qué es un elemento de un conjunto?

Definición

Los objetos de un conjunto se llaman también elementos o miembros del conjunto. Se dice que un conjunto contiene a sus elementos. Cuando objeto es un elemento de un conjunto, se dice que pertenece a dicho conjunto.

Ejemplo

$a \in \{a, e, i, o, u\}$,

Preguntas de interés

- ¿Qué es un elemento de un conjunto?

Definición

Los objetos de un conjunto se llaman también elementos o miembros del conjunto. Se dice que un conjunto contiene a sus elementos. Cuando objeto es un elemento de un conjunto, se dice que pertenece a dicho conjunto.

Ejemplo

$a \in \{a, e, i, o, u\}$,
$-1 \notin \mathbb{N}$,

Preguntas de interés

- ¿Qué es un elemento de un conjunto?

Definición

Los objetos de un conjunto se llaman también elementos o miembros del conjunto. Se dice que un conjunto contiene a sus elementos. Cuando objeto es un elemento de un conjunto, se dice que pertenece a dicho conjunto.

Ejemplo

$a \in \{a, e, i, o, u\}$,
$-1 \notin \mathbb{N}$,
$0 \in \mathbb{Z}$,

Preguntas de interés

- ¿Cómo se puede definir un conjunto?

Preguntas de interés

- ¿Cómo se puede definir un conjunto?
  Por extensión y por comprensión.

Preguntas de interés

- ¿Cómo se puede definir un conjunto?
  Por extensión y por comprensión.

Ejemplo

Por extensión $a \in \{a, e, i, o, u\}$

Preguntas de interés

- ¿Cómo se puede definir un conjunto?
  Por extensión y por comprensión.

Ejemplo

Por extensión $a \in \{a, e, i, o, u\}$
y por comprensión $\{x \mid x \text{ es una vocal}\}$

Preguntas de interés

- ¿Cuándo son iguales dos conjuntos?

Preguntas de interés

- ¿Cuándo son iguales dos conjuntos?

Definición

Dos conjuntos son iguales si, y sólo si, tienen los mismos elementos.

Preguntas de interés

- ¿Cuándo son iguales dos conjuntos?

Definición

Dos conjuntos son iguales si, y sólo si, tienen los mismos elementos.

Axioma de extensión

Si todo elemento de A pertenece a B y todo elemento de B pertenece al conjunto A, entonces los conjuntos son iguales. Escribimos $A = B$.

Preguntas de interés

- ¿Cuándo son iguales dos conjuntos?

Definición

Dos conjuntos son iguales si, y sólo si, tienen los mismos elementos.

Axioma de extensión

Si todo elemento de A pertenece a B y todo elemento de B pertenece al conjunto A, entonces los conjuntos son iguales. Escribimos $A = B$.

Ejemplo

$\{1, 2, 3\} = \{3, 2, 1\}$

Preguntas de interés

- ¿Cuándo son iguales dos conjuntos?

Definición

Dos conjuntos son iguales si, y sólo si, tienen los mismos elementos.

Axioma de extensión

Si todo elemento de A pertenece a B y todo elemento de B pertenece al conjunto A, entonces los conjuntos son iguales. Escribimos $A = B$.

Ejemplo

$\{1, 2, 3\} = \{3, 2, 1\}$
$\{1, 2, 3\} = \{1, 1, 1, 1, 2, 2, 2, 3, 3\}$

Preguntas de interés

- ¿Cuándo son iguales dos conjuntos?

Definición

Dos conjuntos son iguales si, y sólo si, tienen los mismos elementos.

Axioma de extensión

Si todo elemento de A pertenece a B y todo elemento de B pertenece al conjunto A, entonces los conjuntos son iguales. Escribimos $A = B$.

Ejemplo

$\{1, 2, 3\} = \{3, 2, 1\}$
$\{1, 2, 3\} = \{1, 1, 1, 1, 2, 2, 2, 3, 3\}$

Para probar que dos conjuntos son iguales, se debe probar que $A \subseteq B$ y que $B \subseteq A$.

Preguntas de interés

- ¿Qué es un diagrama de Venn?

Preguntas de interés

- ¿Qué es un diagrama de Venn?

Definición

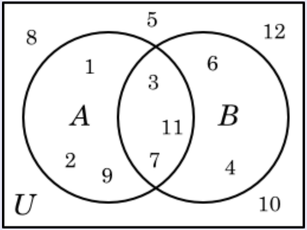
Representación gráfica de conjuntos.

- ¿Qué es un diagrama de Venn?

Definición

Representación gráfica de conjuntos.

Ejemplo



Preguntas de interés

- ¿Qué es la relación de inclusión?

Preguntas de interés

- ¿Qué es la relación de inclusión?

Definición

Se dice que el conjunto A es subconjunto de B, denotado por $A \subseteq B$, si todo elemento de A es elemento de B.

Preguntas de interés

- ¿Qué es la relación de inclusión?

Definición

Se dice que el conjunto A es subconjunto de B, denotado por $A \subseteq B$, si todo elemento de A es elemento de B.

Ejemplo

$\{e, u\} \subseteq \{a, e, i, o, u\}$

Preguntas de interés

- ¿Qué es la relación de inclusión?

Definición

Se dice que el conjunto A es subconjunto de B, denotado por $A \subseteq B$, si todo elemento de A es elemento de B.

Ejemplo

$\{e, u\} \subseteq \{a, e, i, o, u\}$
$\{-1, 0, 1\} \subseteq \mathbb{Z}$

Conjunto vacío

En ocasiones, existen en matemáticas, conjuntos que carecen de elementos.

Conjunto vacío

En ocasiones, existen en matemáticas, conjuntos que carecen de elementos. A este conjunto que carece de elementos se le denomina **Conjunto vacío**. Se puede simbolizar como $\{\}$ o $\emptyset$.

Conjunto vacío

En ocasiones, existen en matemáticas, conjuntos que carecen de elementos. A este conjunto que carece de elementos se le denomina **Conjunto vacío**. Se puede simbolizar como $\{\}$ o $\emptyset$.

La existencia de este conjunto se da como un axioma.

Conjunto vacío

En ocasiones, existen en matemáticas, conjuntos que carecen de elementos. A este conjunto que carece de elementos se le denomina **Conjunto vacío**. Se puede simbolizar como $\{\}$ o $\emptyset$.

La existencia de este conjunto se da como un axioma.

Axioma del conjunto vacío

Existe un conjunto que no tiene elementos

Subconjuntos

Teorema

Para cualquier conjunto S

1. (I) $\emptyset \subseteq S$
2. (II) $S \subseteq S$

Tamaño o número de elementos

Si A es un conjunto, ¿cómo se denomina el número de elementos de un conjunto?

Tamaño o número de elementos

Si A es un conjunto, ¿cómo se denomina el número de elementos de un conjunto?

Definición

*El número de elementos distintos de un conjunto A se denomina **Cardinalidad** de A. Se simboliza como $\#(A)$ o $Car(A)$ o $|A|$.*

Tamaño o número de elementos

Si A es un conjunto, ¿cómo se denomina el número de elementos de un conjunto?

Definición

El número de elementos distintos de un conjunto A se denomina ***Cardinalidad*** *de A. Se simboliza como* $\#(A)$ *o* $Car(A)$ *o* $|A|$.

Ejemplo

$|\{a, e, i, o, u\}| = 5$

Tamaño o número de elementos

Si A es un conjunto, ¿cómo se denomina el número de elementos de un conjunto?

Definición

*El número de elementos distintos de un conjunto A se denomina **Cardinalidad** de A. Se simboliza como $\#(A)$ o $Car(A)$ o $|A|$.*

Ejemplo

$|\{a, e, i, o, u\}| = 5$
$|\{x \mid x \text{ es un dígito}\}| = 10$

Tamaño o número de elementos

Si A es un conjunto, ¿cómo se denomina el número de elementos de un conjunto?

Definición

*El número de elementos distintos de un conjunto A se denomina **Cardinalidad** de A. Se simboliza como $\#(A)$ o $Car(A)$ o $|A|$.*

Ejemplo

$|\{a, e, i, o, u\}| = 5$
$|\{x \mid x \text{ es un dígito}\}| = 10$
$|\emptyset| = 0$

Tamaño o número de elementos

Si A es un conjunto, ¿cómo se denomina el número de elementos de un conjunto?

Definición

El número de elementos distintos de un conjunto A se denomina ***Cardinalidad*** *de A. Se simboliza como* $\#(A)$ *o* $Car(A)$ *o* $|A|$.

Ejemplo

$|\{a, e, i, o, u\}| = 5$
$|\{x \mid x \text{ es un dígito}\}| = 10$
$|\emptyset| = 0$

Definición

Un conjunto finito es un conjunto con una cantidad finita de elementos. De lo contrario se denomina conjunto infinito.