SCC120 Fundamentals of Computer Science

Haiyang LIU (刘海洋)

School of Computer and Information Technology
haiyangliu@bjtu.edu.cn
BJTU-Weihai-SD404

SCC120 Delivery Plan

- Ten Weeks: Week 1 — Week 10
- Weekly Plan:
  --Lecture (2h); Workshop (2h)
- Assessment Model
  --20% In-class Test; 20% Coursework; 60% next term

SCC120 Delivery Plan

- Discrete Mathematics (Week 1—Week 2)
- Data Structures (Week 3—Week 6)
- Algorithms (Week 7—Week 10)

SCC120 Delivery Plan

- **In-class Test**: test takes place in Week 6. Immediate feedback (you get the answers and commentary on the same or the next day) afterwards.

- **Coursework**: out week 9, due week 11. Marks and feedback are expected at the end of this term. This will be submitted online and checked for plagiarism automatically.

Discrete Maths : 1

SCC120 Fundamentals of
Computer Science

Discrete Maths

- What? : The study of objects that are discrete rather than continuous.
  - Real numbers : vary “smoothly” – continuous.
  - Discrete maths objects : do not vary smoothly, but have discrete, separated values i.e. integers, graphs, logic statements
- Why? : Foundation for formal methods:
  - mathematical approaches to software and hardware computer-based systems
  - software engineering and software testing

Syllabus : DM1

- Sets
- Relations
- Functions
- Recursion Recursion Recursion Recursion

DM2 : Logic

Syllabus : DM1

- Sets
- Relations
- Functions
- Recursion Recursion Recursion Recursion

Overview

- Sets
  - Defining sets
  - Set operations
  - Types of sets
- Objectives
  - Understanding the basic ideas about sets
  - Facility with set operations

Sets and membership

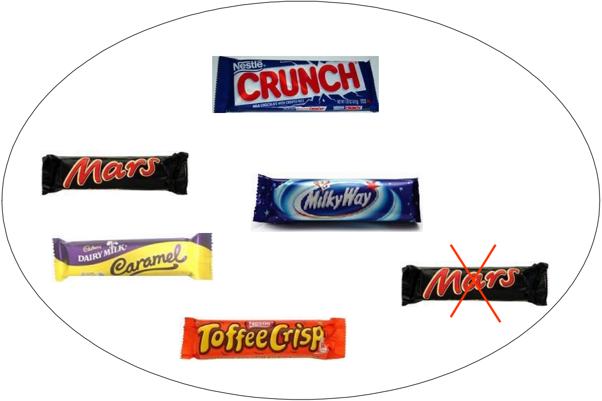
- Set = collection of objects
  - in a set there are **no duplicates**
  - a set is **unordered**
- Example set: $A = \{1, 2, 3, 4, 5, 6, 7\}$
- Set membership is notated using the symbol $\in$
- 1 is in set A:
  - 1 belongs to the set A
  - 1 is an element/object/member of the set A
  - written as: $1 \in A$
  - 1 is not in set B: $1 \notin B$ (1 does not belong to set B)
- Sets named using single capital letter.

Chocolate Bars

- Other chocolate bars are available.
- Other manufacturers exist.

The Set of Chocolate Bars

No duplicates allowed!



Order does not matter

Sets are unordered

Identical sets

Defining sets

- Listing all its members-- Enumeration
  - writing down all the elements
  - A = {a, b, {a, b}, c}
  - small, finite sets

- Listing a property that its members must satisfy
- {x is an integer | 0 < x < 8 }
  - every integer x such that x is greater than 0 and less than 8
- infinite sets: {x is an real number| x > 0}
  - every real number x such that x is greater than 0

SET OPERATIONS

Set operations

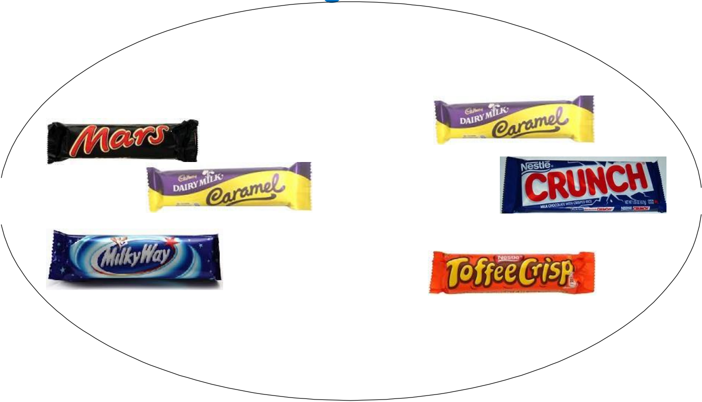
- Union
- Intersection
- Difference
- Cartesian product

SET OPERATIONS : UNION

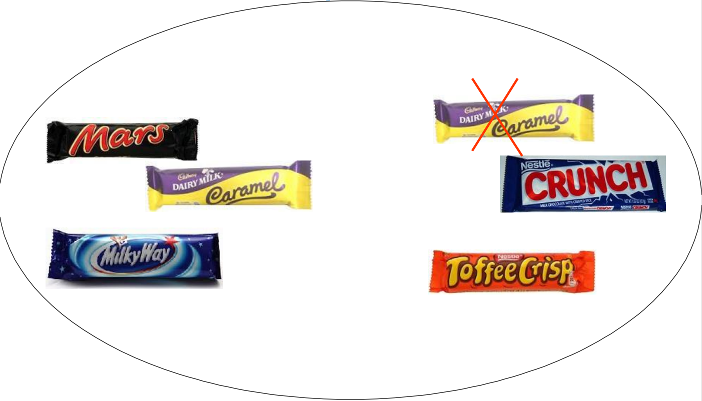
Set operations: Union ($\cup$)

$$C = A \cup B$$

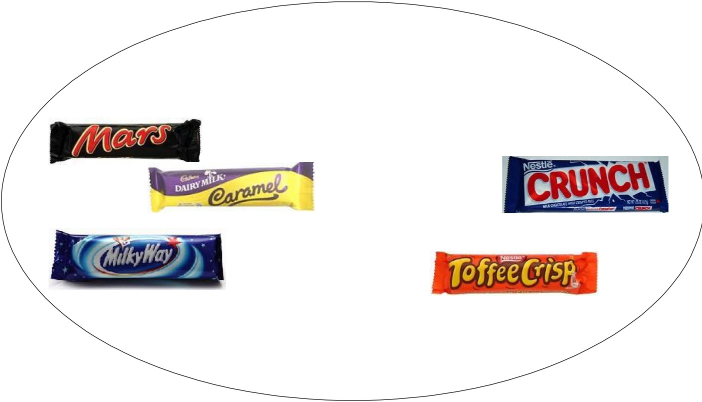
Building a Union



No duplicates!



Final result : C = A ∪ B



Set Operations

UNION (written $\cup$)

- forms a new set from two sets consisting of all elements that are in EITHER of the original sets (with no duplicate elements)
- $A \cup B = \{ x \mid x \in A \textbf{ or } x \in B \}$

- Examples
  - If $A = \{1, 2, 3\}$ and $B = \{4, 5\}$
  - $A \cup B = \{1, 2, 3, 4, 5\}$

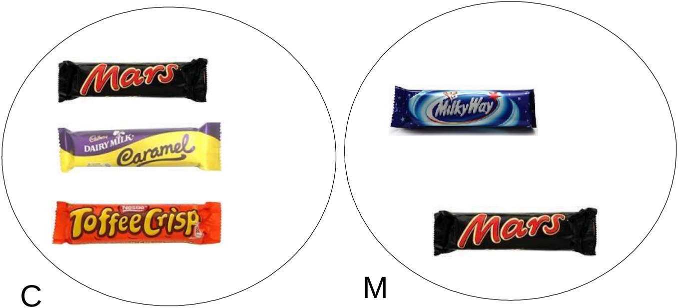
  - If $A = \{1, 2, 3\}$ and $B = \{1, 2, 4, 5\}$
  - $A \cup B = \{1, 2, 3, 4, 5\}$

What elements do the sets have in common?

SET OPERATIONS : INTERSECTION

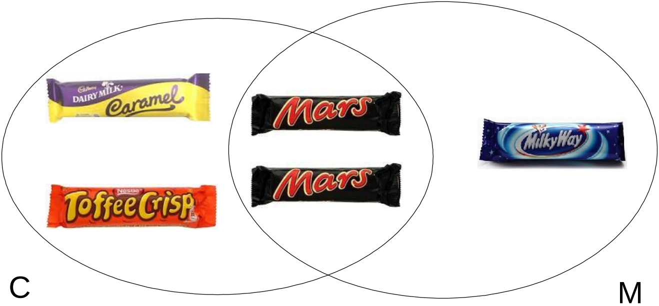
Two Sets

Let set C contain bars that contain caramel.
Let set M contain bars that contain nougat.



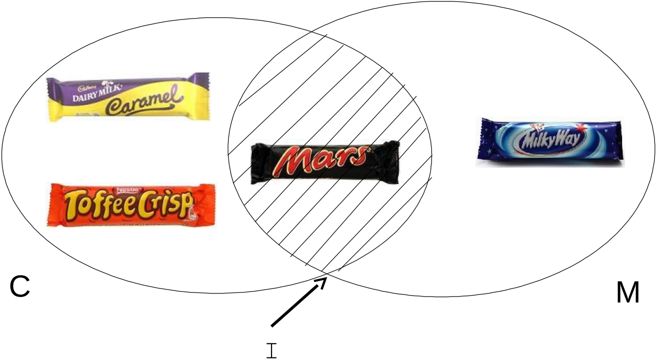
Building an Intersection

Building an Intersection



No duplicates!

Final Result : $I = C \cap M$



Using shading to indicate result of operation

Set Operations

INTERSECTION (written $\cap$)

- forms a new set from two sets, consisting of all elements that are in BOTH of the original sets
- $A \cap B = \{ x \mid x \in A \text{ and } x \in B \}$

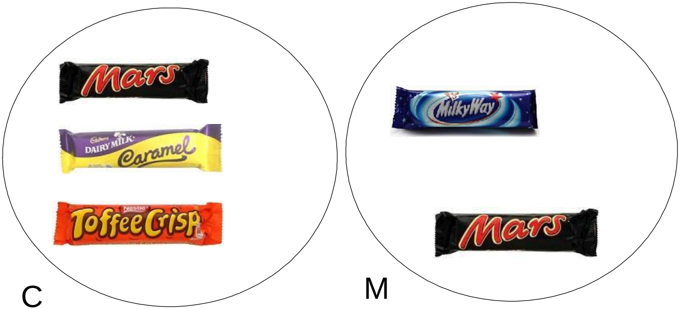
- Examples
  - If $A = \{1, 2, 3\}$ and $B = \{1, 2, 4, 5\}$
  - $A \cap B = \{1, 2\}$

  - If $A = \{1, 2, 3\}$ and $B = \{4, 5\}$
  - $A \cap B = \varnothing$

What does one set contain that the other one doesn’t?

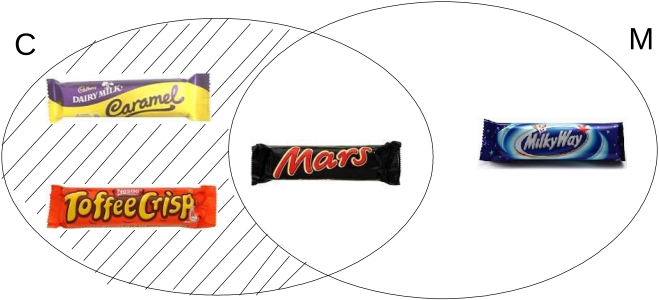
SET OPERATIONS : DIFFERENCE

Set operations: Difference (-)



D = C - M

Set operations: Difference (-)



D = C - M

Set Operations

DIFFERENCE (written – or /)

- forms a new set from two sets, consisting of all elements from the first set that are not in the second
- $A - B = \{ x \mid x \in A \textbf{ and } x \notin B \}$

- Examples
  - If $A = \{1, 2, 3\}$ and $B = \{1, 2, 4, 5\}$
  - $A - B = \{3\}$

  - If $C = \{1, 2, 3\}$ and $D = \{4, 5\}$
  - $C - D = \{1, 2, 3\}$
- What do B – A and D – C contain?

SET OPERATIONS : CARTESIAN PRODUCT

Cartesian product

Ordered pair

- is a pair of objects with an order associated with them
- If objects are represented by x and y, then we write the **ordered pair as $\langle x, y \rangle$**
- Two ordered pairs $\langle a, b \rangle$ and $\langle c, d \rangle$ are equal if and only if $a = c$ and $b = d$.
- Example the ordered pairs
  - $\langle 1, 2 \rangle$ and $\langle 2, 1 \rangle$ are not equal.

Cartesian product : A x R

- A = { a1, a2, a3, a4, a5 }
- R = { r1, r2, r3, r4 }

	r1	r2	r3	r4
a1	<a1, r1>	<a1, r2>	<a1, r3>	<a1, r4>
a2	<a2, r1>	<a2, r2>	<a2, r3>	<a2, r4>
a3	<a3, r1>	<a3, r2>	<a3, r3>	<a3, r4>
a4	<a4, r1>	<a4, r2>	<a4, r3>	<a4, r4>
a5	<a5, r1>	<a5, r2>	<a5, r3>	<a5, r4>

Cartesian product

Cartesian product of *A* and *B*

- The set of all ordered pairs <*a, b*>
  - where *a* is an element of *A* and *b* is an element of *B*
- written *A* x *B*.

Example:

A = {1, 2, 3} and *B* = {a, b}. Then

- *A* x *B* = {<*1, a*>, <*1, b*>, <*2, a*>, <*2, b*>, <*3, a*>, <*3, b*>}
- *B* x *A* = {<*a, 1*>, <*a, 2*>, <*a, 3*>, <*b, 1*>, <*b, 2*>, <*b, 3*>}

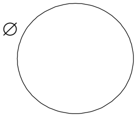
TYPES OF SETS

Types of sets

- Empty set
- Subset, proper subset
- Universal set, complement set
- Standard sets

Empty set

- The **empty set** is the set which contains no objects
- Denoted by the symbol $\varnothing$



Subset

- If A and B are sets, and every element of A is also an element of B, then
- A is a subset of (or is included in) B.
- $A \subseteq B$
- $C = \{1, 2\}$ and $D = \{1, 2, 3\}$
- $C \subseteq D$

Proper Subset

- If A is a subset of B, but B is not equal to A (i.e. there exists at least one element of B that is not contained in A), then

- A is also a proper (or strict) subset of B
- $A \subset B$

Proper Subset Example

- C = {1, 2} and D = {1, 2, 3}
- The set C is a proper subset of D because
  - C is a subset of D (all the elements of C are also contained in D)
  - D contains at least one element (3) which is not contained in C.
- $C \subset D$
- Any set is a subset of itself, but not a proper subset. Why not?

Subset Exercises

- if $A = \{a, b, c, d, e, f\}$, $B = \{a, b, e\}$, $C = \{c, d\}$, and $D = \{d, f, g\}$ say which of the following are true statements:

1. $B \subseteq B$
2. $B \subset B$
3. $B \subset A$
4. $C \subseteq A$
5. $(B \cup C) \subset A$
6. $D \subseteq A$

1. Answer: true (of any set)
2. Answer: false (of any set)
3. Answer: true
4. Answer: true
5. Answer: true
6. Answer: false

Universal sets

- A non-empty set of which all the sets under consideration are subsets is called the universal set.
- Usually denoted by U.
- Example: set of real numbers R is a universal set for the operations related to real numbers

Complement sets

The **complement set** is the difference between the universe and a given set

- Denoted: comp(A) = U - A

Example: U = {a, b, c, d, e, f, g}, A = {a, b, c} and B = {b, c, d, e}

- comp(A) = {d, e, f, g}
- comp(B) = {a, f, g}
- comp ($A \cup B$) = comp({a, b, c, d, e}) = {f, g}

Some Standard Sets

- **N** : all natural numbers
  N = { 1, 2, 3, 4, ...}
  Sometimes includes 0.
- **Z** : all integers
  Z = { ..., -2, -1, 0, 1, 2, ...}
- **R** : all real numbers

Syllabus : DM1

- Sets
- Relations
- Functions
- Recursion Recursion Recursion Recursion

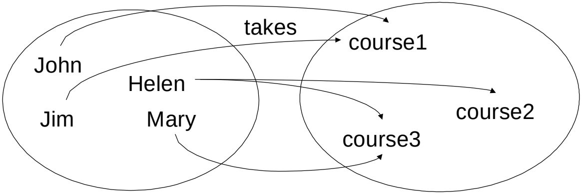
Objectives

- Understanding the basic ideas about relations
- Ability to represent relations

Overview

- Preliminary
  - – Ordered pairs
  - – Cartesian product
- Binary and n-ary relations
  - – Definitions
  - – Equality of relations

Association



John takes course1, Jim takes course1, Mary takes course3, Helen takes courses 2 and 3.

- sets of related objects (John, course1)
- order matters

- A = { Helen, Jim, John, Mary }
- B = { course1, course2, course3}
- 4 * 3 elements = 12 pairs

< Helen, course1 >
< Helen, course2 >
< Helen, course3 >
< Jim, course1 >
< Jim, course2 >
< Jim, course3 >
< John, course1 >
< John, course2 >
< John, course3 >
< Mary, course1 >
< Mary, course2 >
< Mary, course3 >

Associations

- John takes course1, Jim takes course1, Mary takes course3,
- Helen takes courses 2 and 3.

< Helen,	course1 >
< Helen,	**course2 >**
< Helen,	**course3 >**
< Jim,	**course1 >**
< Jim,	course2 >
< Jim,	course3 >
< John,	**course1 >**
< John,	course2 >
< John,	course3 >
< Mary,	course1 >
< Mary,	course2 >
< Mary,	**course3 >**

Definitions

- Binary relation R from a set A to a set B
  - a set of ordered pairs <a, b> , $a \in A$ and $b \in B$
- an ordered pair <a, b> is in a relation R
  - element a is related to element b in relation R
  - Written: a R b, or $\langle a, b \rangle \in R$
  - If A = B, the relation from A to B becomes relation on A
- what is the relationship between R and A x B?
  - A x B is the set of all ordered pairs <a, b>
  - R is a subset of A x B: $R \subseteq A \times B$

Example: Binary relations

- Ordered pairs
  - <John, course1>
  - <Jim, course1>
  - <Mary, course3>
  - <Helen, course2>
  - <Helen, course3>
- Relation: *takes*
  - John takes course1; < John, course1> ∈ *takes*
  - Jim takes course1; < Jim, course1> ∈ *takes*
  - Mary takes course3; <Mary, course3> ∈ *takes*

Definitions : Tuples

- In mathematics and computer science a **tuple** captures the intuitive notion of an ordered list of elements.
- Ordered n-tuple
  - on n sets A1, A2, …, An.
  - ordered n-tuple is a set of n objects with an order associated with them
    - written: **<x1, x2, ..., xn>**.
    - n sets and n elements in the n-tuple
    - x1 $\in$ A1, x2 $\in$ A2, ... xn $\in$ An.

- Example: A = {John, Jim, Helen, Mary},
  B = {course1, course2, course3},
  C = {65, 41, 55, 72, 63}
- Some ordered n-tuples on sets A, B, C
- <John, course1, 65>,

  <Jim, course1, 41>,

  <Mary, course3, 55>,

  <Helen, course2, 72>,

  <Helen, course3, 63>

Definitions

- Equality of n-tuples:
  - <x1, ..., xn> = <y1, ..., yn> if and only if
  - x1 = y1, x2 = y2, x3 = y3, …, xn = yn

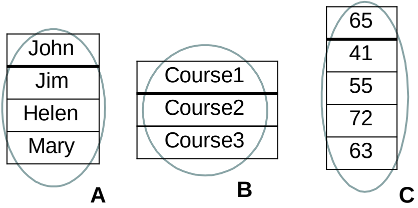
    ($x_i = y_i$ for all i, $1 \leq i \leq n$)

  - Example: the ordered 3-tuple
  - <1, 2, 3> = <1, 2, 3> and
    <1, 2, 3> ≠ <2, 3, 1>

Definitions

- Cartesian product of n sets A1, ..., An
  - the set of all possible ordered n-tuples <x1, x2, ... , xn> , where $x1 \in A1, x2 \in A2, \ldots, xn \in An$ ($xi \in Ai$, for all i, $1 \le i \le n$)
  - written: A1 x A2 x... x An

- How many n-tuples in A X B X C?
- Every possible combination of the values.
- 4 * 3 * 5 = 60.



Example

- Let $A = \{0, 1, 2, 3\}$ and $B = \{0, 1, 2, 3, 4\}$.
- List the ordered pairs in the relation R from A to B where $\langle a, b\rangle \in R$ if and only if $b - a = 1$

Answer:

for $a = 0$, what is the value of b?

$b - a = 1$ means $b = a + 1$

or $\langle a, b\rangle = \langle a, a + 1\rangle$

$R = \{\langle 0, 1\rangle, \langle 1, 2\rangle, \langle 2, 3\rangle, \langle 3, 4\rangle\}$

Equality of relations

- Binary relations:
- $R1 \subseteq A1 \times A2$ and $R2 \subseteq B1 \times B2$
- When are two relations equal?
- R1 = R2 if and only if
  - the same sets: A1 = B1 , A2 = B2;
  - the set of things related are the same: R1 = R2 as sets
- Ex. $R1 = \{\langle 1, 2\rangle , \langle 2, 2\rangle\} \subseteq \{1, 2\} \times \{1, 2\}$,
- $R2 = \{\langle a, b\rangle , \langle b, b\rangle\} \subseteq \{a, b\} \times \{a, b\}$.
- R1 = R2 if and only if a = 1 and b = 2.

Equality of relations

- n-ary relation $R1 \subseteq A1 \times \ldots \times An$
- m-ary relation $R2 \subseteq B1 \times \ldots \times Bm$
- $R1 = R2$ if and only if
  - $m = n$,
  - $Ai = Bi$ for each i, $1 \leq i \leq n$, and
  - $R1 = R2$ as a set of ordered n-tuples

Syllabus : DM1

- Sets
- Relations--Diagraphs
- Functions
- Recursion Recursion Recursion Recursion

DIAGRAPHS

Diagraphs

- Directed graph

- A diagram composed of:
  - points (i.e. vertices, nodes)
  - arrows (i.e. arcs) which connect points to other points

- Diagraph is an ordered pair of sets $G = (P, A)$:
  - P is a set of points
  - A is a set of ordered pairs (called arcs) of points of P.

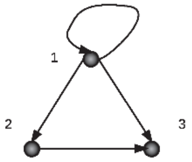
Diagraphs

Example

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

$A = \{<1, 1>, <1, 2>, <1, 3>, <2, 3>\}$

G1 = (P, A)



G1

Representing binary relations through diagraphs

- $R \subseteq P \times P$
- elements of set P are points of the diagraph G
- <p1, p2> is an arc of G from point p1 to point p2 if and only if <p1, p2> is in R

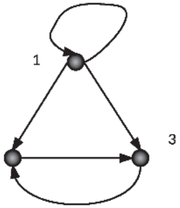
Diagraphs

Example

P = { *1, 2, 3* }

A = {<*1, 1*>, <*1, 2*>, <*1, 3*>, <*2, 3*>, <3,2> }

G1 = (P, A)



G1

If you want a bi-directional arc, say from point 3 to point 2, you need to add the ordered pair <3,2> to set A.

Examples

Draw the diagraphs of the following relations on the set $A = \{1, 2, 3, 4\}$

- equal (=)
- less than (<)
- different ($\neq$)

Equals

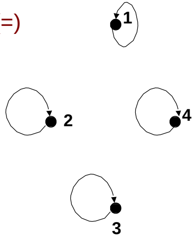
	1	2	3	4
1	**<1, 1>**	<1, 2>	<1,3>	<1,4>
2	<2, 1>	**<2,2>**	<2,3>	<2,4>
3	<3,1>	<3,2>	**<3,3>**	<3,4>
4	<4,1>	<4,2>	<4,3>	<4,4>

<1, 1>			
	<2,2>		
		<3,3>	
			<4,4>

Equals

- $A = \{1, 2, 3, 4\}$
- R = {<*1*, *1*> , <2, *2*>, <3, 3>, <4, *4*> }

(=)



Less Than

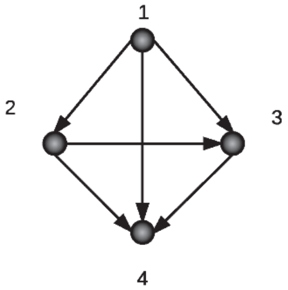
	1	2	3	4
1	<1, 1>	**<1, 2>**	**<1,3>**	**<1,4>**
2	<2, 1>	<2,2>	**<2,3>**	**<2,4>**
3	<3,1>	<3,2>	<3,3>	**<3,4>**
4	<4,1>	<4,2>	<4,3>	<4,4>

<1, 2>	**<1,3>**	**<1,4>**
	<2,3>	**<2,4>**
		<3,4>

Less Than

- A = {1, 2, 3, 4}
- R = {<1, 2> , <1, 3>, <1, 4>, <2, 3> , <2, 4> , <3, 4> }

(<)



different (≠)

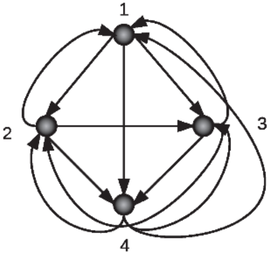
- The values in the ordered pairs participating in this relation should be different.
- <a, b> such that a ≠ b.

	1	2	3	4
1	**<1, 1>**	<1, 2>	<1,3>	<1,4>
2	<2, 1>	**<2,2>**	<2,3>	<2,4>
3	<3,1>	<3,2>	**<3,3>**	<3,4>
4	<4,1>	<4,2>	<4,3>	**<4,4>**

<1, 2>	<1,3>	<1,4>
<2, 1>	<2,3>	<2,4>
<3,1>	<3,2>	<3,4>
<4,1>	<4,2>	<4,3>

different (≠)

R = {<1, 2>, <1, 3>, <1, 4>,
<2, 1>, <2, 3>, <2, 4>,
<3, 1>, <3, 2>, <3, 4>,
<4, 1>, <4, 2>, <4, 3>}



Properties of Diagraphs

- Reflexivity
- Irreflexivity
- Symmetry
- Transitivity

Reflexivity *Property*

$R \subseteq A \times A$ is **reflexive** if and only if

$\langle a, a \rangle \in R$ for every element a of A

- every element of A is in relation with itself

- Consider B = { 5, 6, 7}.
- For R to be reflexive, it must contain
- < 5, 5 >, < 6, 6> and <7, 7>
- It can contain other elements as well, but it must have these three.
- If it has less than these three i.e. <5,5> and <7,7>, then that is insufficient for reflexivity.

Less Than Or Equal

- <a,b> such that a <= b

	1	2	3
1	**<1, 1>**	**<1, 2>**	**<1,3>**
2	<2, 1>	**<2,2>**	**<2,3>**
3	<3,1>	<3,2>	**<3,3>**

Reflexivity *Property*

Example: “<=” on A={1, 2, 3} is { **<1, 1>,** <1, 2>, <1, 3>, **<2, 2>**, <2, 3>, **<3, 3>** } and it is reflexive because ***<1, 1>, <2, 2>, <3, 3>*** are in this relation.

Irreflexivity *Property*

Irreflexivity

$R \subseteq A \times A$ is **irreflexive** if and only if

$\langle a, a \rangle \notin R$ for every element a of A.

- **no** element of A is in relation with itself

Less Than

- < a,b > such that a < b
- LT = { <1, 2>, <1,3>, <2,3> }

	1	2	3
1	<1, 1>	**<1, 2>**	**<1,3>**
2	<2, 1>	<2,2>	**<2,3>**
3	<3,1>	<3,2>	<3,3>

Greater Than

- < a, b > such that a > b
- GT = { <2,1>, <3, 1>, <3, 2> }

	1	2	3
1	<1, 1>	<1, 2>	<1,3>
2	**<2, 1>**	<2,2>	<2,3>
3	**<3,1>**	**<3,2>**	<3,3>

Irreflexivity Property

- Example: “>” and “<“ on the set of integers
  {1, 2, 3} are irreflexive.
- LT = { <1, 2>, <1,3>, <2,3> }
- GT = { <2,1>, <3, 1>, <3, 2> }
- Neither of these sets contain any elements where <a, b> such that a = b.

Symmetry *Property*

Symmetry

$R \subseteq A \times A$ ***is*** **symmetric** if and only if
for any ***a***, and ***b*** in ***A***,
whenever **<*a, b*>** $\in$ ***R*** then **<*b, a*>** $\in$ ***R*****.**

Symmetry *Property*

Example: “=“ on the set of integers ***{1, 2, 3}*** is ***{<1, 1> , <2, 2> <3, 3> }*** and it is symmetric.

EQ = { <1,1>, <2, 2>, <3, 3> }

	1	2	3
1	**<1, 1>**	<1, 2>	<1,3>
2	<2, 1>	**<2,2>**	<2,3>
3	<3,1>	<3,2>	**<3,3>**

Symmetry *Property*

If R = A x A, then R is symmetric.

	1	2	3
1	<1, 1>	<1, 2>	<1,3>
2	<2, 1>	<2,2>	<2,3>
3	<3,1>	<3,2>	<3,3>

<1,1>	**<1,1>**
<2,2>	**<2,2>**
<3,3>	**<3,3>**
<1,2>	**<2,1>**
<1,3>	**<3,1>**
<2,3>	**<3,2>**

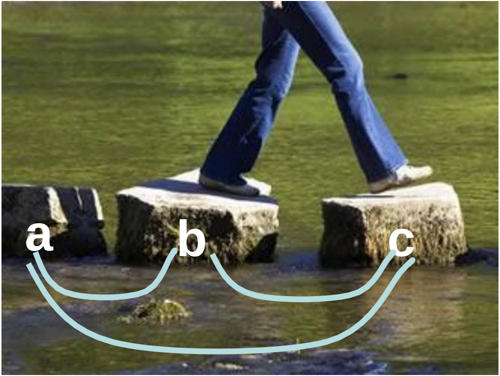
Symmetric Pairs

Transitivity Property

$R \subseteq A \times A$ is transitive if and only if
for any a, b, and c $\in$ A,
if $\langle a, b \rangle \in R$, and $\langle b, c \rangle \in R$
- then $\langle a, c \rangle \in R$

- If you can get from a to b, and from b to c,
- Then you can get from a

Stepping Stones



Transitivity Property

Example1: “≤”on the set {1, 2, 3} is transitive, because for

<1, 2> and <2, 3> in “≤”, then

<1, 3> is also in “≤”

for <1, 1> and <1, 2> in “≤”,

then <1, 2> is also in “≤”

- and similarly for the others

	1	2	3
1	**<1, 1>**	**<1, 2>**	**<1,3>**
2	<2, 1>	**<2,2>**	**<2,3>**
3	<3,1>	<3,2>	**<3,3>**

Transitivity Property

- Example : "is the mother of" is not a transitive relation, because if Alice is the mother of Brenda, and Brenda is the mother of Claire, then Alice is not the mother of Claire.

$b = a + 1$

- { <1,2>, <2,3>, <3,4> }
- Non-transitive

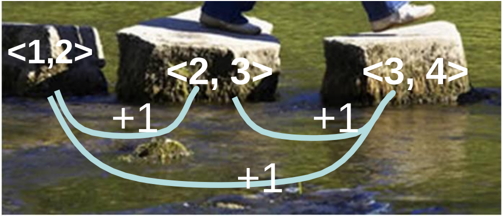
1 + 1 = 2,

2 + 1 = 3,

therefore

1 + 1 = 3

???????



Properties : Equivalence relation

- $R \subseteq A \times A$ is an equivalence relation if and only if:
  - R is reflexive, and
  - R is symmetric, and
  - R is transitive

  - Example : The equality relation “=“ is an equivalence relation

Syllabus : DM1

- Sets
- Relations
- Functions
- Recursion Recursion Recursion Recursion

Objectives

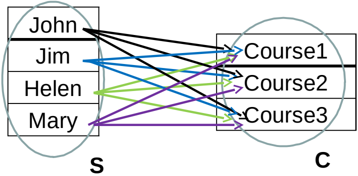
- Understanding the basic ideas about functions
- Ability to operate with functions

Overview

- Preliminary
  - Cartesian product
  - Relations
- Functions
  - Definitions
  - Operations
  - Types

Preliminary

Let S = {John, Jim, Helen, Mary}
and C = {course1, course2, course3}



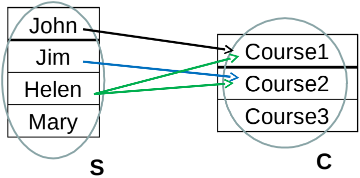
Cartesian product:

S x C = {<John,course1>, <John,course2>, <John,course3>, <Jim,course1>, <Jim,course2>, <Jim,course3>, <Helen,course1>, <Helen,course2>, <Helen,course3>, <Mary,course1>, <Mary,course2>, <Mary,course3>}

Each student takes **each** course (**all possible** ordered pairs)

Preliminary

Let S = {John, Jim, Helen, Mary}
and C = {course1, course2, course3}



Relation: takes $\subseteq$ S x C

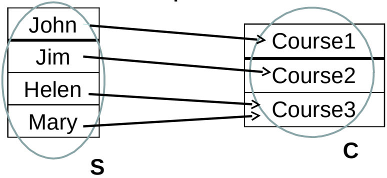
takes = {<John, course1>, <Jim, course2>,
<Helen, course1>, <Helen, course2>}

Some students take some courses

(some ordered pairs)

Definition

- A function is a special type of binary relation. It associates each element of a set with a unique element of another set.



this is a function

Function1 = {<John, Course1>, <Jim, Course2>, <Helen, Course3>, <Mary, Course3>}
is a function from S to C.

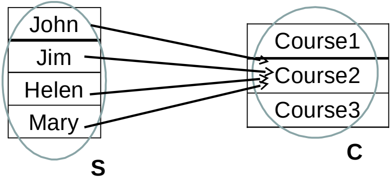
f1(John) = Course1 f1(Jim) = Course2
f1(Helen) = Course3 f1(Mary) = Course3

Each student takes exactly one course.

One course can be taken by many students.

Definition

- A function is a special type of binary relation. It associates each element of a set with a unique element of another set.



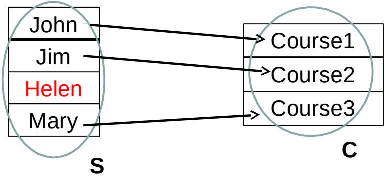
this is a function

Function2 = {<John, Course2>, <Jim, Course2>, <Helen, Course2>, <Mary, Course2>}
is a function from S to C.

f2(John) = Course2 f2(Jim) = Course2
f2(Helen) = Course2 f2(Mary) = Course2

Definition

- A function is a special type of binary relation. It associates each element of a set with a unique element of another set.

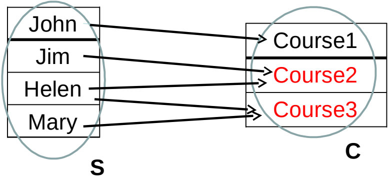


this is NOT a function; but it is a relation

Illegal : f1(Helen) = ???
Helen is not associated with any element in set C

Definition

- A function is a special type of binary relation. It associates each element of a set with a unique element of another set.



this is NOT a function; but it is a relation

Illegal : f1(Helen) = { Course2, Course3 }
Helen is associated with more than **one** element in set C

Definition

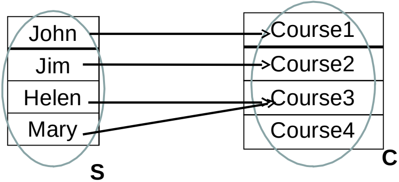
- A function from a set A to a set B is a relation from A to B that satisfies:
  - for each element a in A, there is an element b in B such that <a, b> is in the relation, and
  - if <a, b> and <a, c> are in the relation, then b = c.

- Also said
- for every a $\in$A, there exists a unique b $\in$ B such that a f b, or equivalently <a, b> $\in$ f.

Definition

- We write a function f from S to C:
- ***f*** : S $\rightarrow$ C

- The set S is called the domain of f and the set C is called the codomain of f



- domain of f = S = {John, Jim, Helen, Mary}
- codomain of f = C = {Course1, Course2, Course3, Course4}
- f = {<John, Course1>, <Jim, Course2>, <Helen, Course3>, <Mary, Course3>}

Definition

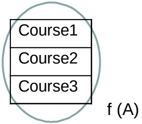
- f = {<John, Course1>, <Jim, Course2>, <Helen, Course3>, <Mary, Course3>}
- If <a, b> ∈ f, then b is denoted by f(a):
  - b = f(a)
  - b is the image of a under f
  - a is the preimage of b under f

b	image of	a	under f
Course1	*image of*	John	*under f*
Course2	*image of*	Jim	*under f*
Course3	*image of*	Helen	*under f*
Course3	*image of*	Mary	*under f*

a	preimage of	b	under f
John	*preimage of*	Course1	*under f*
Jim	*preimage of*	Course2	*under f*
Helen	*preimage of*	Course3	*under f*
Mary	*preimage of*	Course3	*under f*

Definition

b	image of	a	under f
Course1	*image of*	John	*under f*
Course2	*image of*	Jim	*under f*
Course3	*image of*	Helen	*under f*
Course3	*image of*	Mary	*under f*

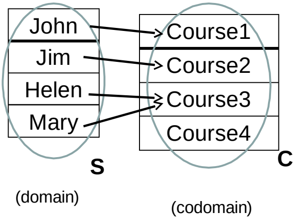


- The range of f is the set of images of f (the set f (A)).
- The range of f and the codomain of f are not necessarily equal.

range of f = { Course1, Course2, Course3}
codomain of f = { Course1, Course2, Course3, Course4}

Example

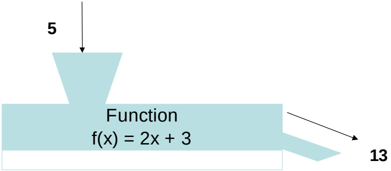
- In terms of the arrows shown in the diagram, note that
  - **every** element of S must have an arrow leaving it
  - **not** every element of C has to be pointed at
- Hence the range of the function {C1, C2, C3} is not necessarily equal to the codomain {C1, C2, C3, **C4**} .



ALGEBRAIC FUNCTIONS

Functions

- Function as a machine
- Each different number going into the machine has one and only one corresponding number that comes out.
- The machine calculates:
- $f(5) = 2x+3 = 2 * 5 + 3 = 10 + 3 = 13$



Defining/declaring a function

Call this the "function header"

$$\boldsymbol{f}\,(x) = 3x + 1$$

Call this the "function body"

The Formal Parameter

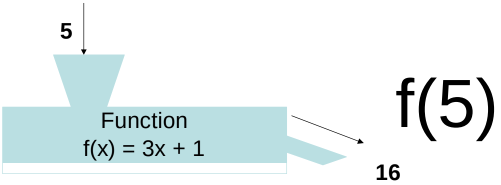
This is the formal parameter …

$$\boldsymbol{f}\,(\,x\,) = 3\,x + 1$$

… which can be referenced in the function body

Calling a function

- When we call a function, we have to supply an **actual parameter**.
- This will be the value that the **formal parameter** takes when the function is executed.



Using expressions as actual parameters

- f (x) = x + 4
- f(6 + 10) = 6 + 10 + 4 = 20
- g (x) = x * 2
- g(6 + 10) = (6+10) * 2 = 16 * 2 = 32

f(	x	) =	x	+	4
f(	6+10	) =	(6+10)	+	4

g(	x	) =	x	*	2
g(	6+10	) =	(6+10)	*	2

Using expressions as actual parameters

- definition : f (x) = x + 4
- function call : f (y) = y + 4
- definition : g (x) = x * 2
- function call : g(y + 2) = (y+2) * 2 = 2y + 4

f(	x	) =	x	+	4
f(	y	) =	y	+	4

g(	x	) =	x	*	2
g(	y + 2	) =	(y + 2)	*	2

Expanding a function call

- f (x) = x + 4
- g (x) = x * 2
- f (g(5)) = f (5 * 2) = f(10) = 10 + 4 = 14

g(	x	)=	x	*	2		
g(	5	)=	5	*	2		
		f(	5	*	2	)	
		f(	10	)			
		f(	x	)=	x	+	4
		f(	10	)=	10	+	4

Expanding a function call

- f (x) = x + 4
- g (x) = x * 2
- g (f(5)) = g (5 + 4) = g (9) = 9 * 2 = 18

f(	x	)=	x	+	4		
f(	5	)=	5	+	4		
		g(	5	+	4	)	
		g(	9	)			
		g(	x	)=	x	*	2
		g(	9	)=	9	*	2

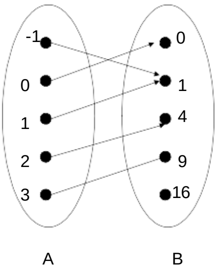
Example

- $A = \{-1, 0, 1, 2, 3\}$
- $B = \{0, 1, 4, 9, 16\}$

- $f : A \to B$
- $f(x) = x^2$

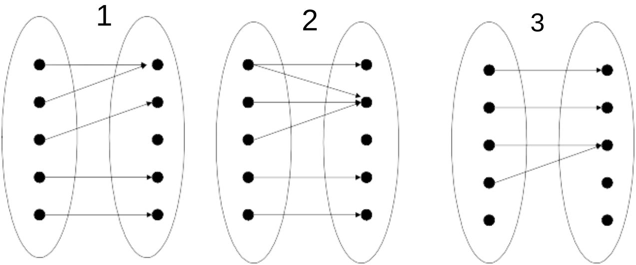
$$f(-1) = (-1)^2 = 1$$
$$f(0) = 0^2 = 0$$
$$f(1) = 1^2 = 1$$
$$f(2) = 2^2 = 4$$
$$f(3) = 3^2 = 9$$

- Range = $\{0, 1, 4, 9\}$

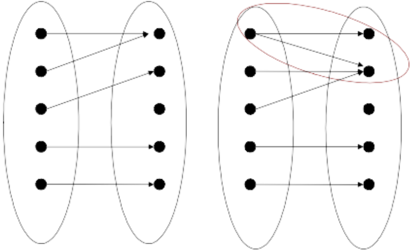


Exercise

Which one is a function?



Answer



OPERATIONS

Addition

Let $f : A \to R$ and $g : A \to R$.

- Sum of f and g :
  - $(f + g)(x) = f(x) + g(x)$, for all x in A

- Example:
  - $f(x) = 3x + 1$ and $g(x) = x^2$
  - $(f + g)(x) = 3x + 1 + x^2 = x^2 + 3x + 1$
- "simply" take the function body of f and add it to the function body of g.

Multiplication

Let $f : A \to R$ and $g : A \to R$.

- Product of f and g :
  - $(f * g)(x) = f(x) * g(x)$, for all x in A
- Example:
  - $f(x) = 3x + 1$ and $g(x) = x^2$
  - $(f * g)(x) = f(x) * g(x) = (3x + 1) * x^2$
    $= x^2 * (3x + 1)$
    $= (x^2 * 3x) + (x^2 * 1)$
    $= 3x^3 + x^2$
- "Simply" take the body of f and multiply it by the body of g.

Composite function

- Let $g : A \to B$, and $f : B \to C$.
- The composition of functions f and g is written as $f \circ g : A \to C$
and is

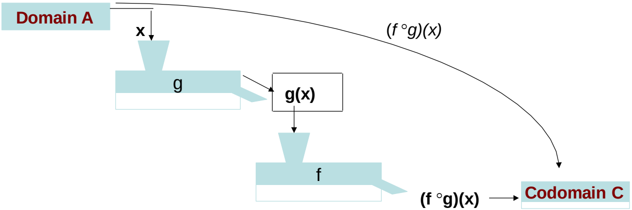
$$(f \circ g)(x) = f(g(x)) \quad \text{for all } x \text{ in } A$$

- Note: the function g is applied first and then f.

Composition

- A composite function is a function within a function
- Let $g : A \rightarrow B$, and $f : B \rightarrow C$
- $f \circ g : A \rightarrow C$ is $(f \circ g)(x) = f(g(x))$ for all x in A

- In terms of "function machines", the composition $f \circ g$ is the function which feeds an input to g and feeds the output of g to f



Composition Example

Let $g : A \to B$ with $g(x) = x + 1$ and
$f : B \to C$ with $f(x) = 2x$

Then $(f \circ g)(x) =$
$f(\,g(x)\,) = f(x+1) = 2\,(\,x + 1\,)$

Syllabus : DM1

- Sets
- Relations
- Functions --- Inverse Functions

Recursion

Recursion

- Recursion Recursion

Inverse Functions

- In the case where every element of A is linked to exactly one element of B, and
- every element of B is linked to exactly one element of A,

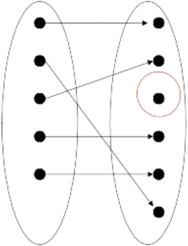
we can have an inverse function.

Bijective Function

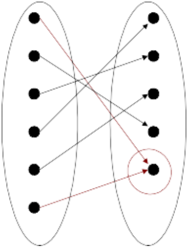
- “A **bijection** (or **bijective function** or **one-to-one correspondence**) is a function giving an *exact* pairing of the elements of two sets.
- Every element of one set is paired with exactly one element of the other set, and every element of the other set is paired with exactly one element of the first set.
- There are no unpaired elements.”

Examples

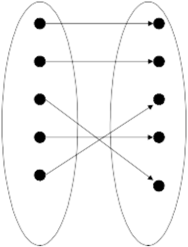
1 Not bijective



2 Not bijective



3 Bijection



Inverse function

- Every bijection has a function called the inverse function.
- Let f be a bijection, $f : A \rightarrow B$.

- $g : B \rightarrow A$ is called the inverse function of f, if for every element y of B, $g(y) = x$, whenever $f(x) = y$.

- Note that such an x is unique for each y because f is a bijection.

- The function g(x) is the inverse function of f(x) and is denoted by f^{-1}

Inverse function

Let $f : Z \to Z$, $f(x) = 2x + 5$

f is bijective. Its inverse is $f^{-1}(x)$

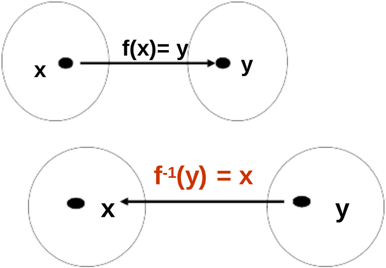
$f(x) = 2x + 5$

$y = 2x + 5$

$2x = y - 5$

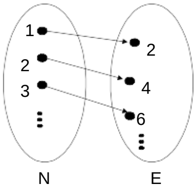
$x = (y - 5)/2$

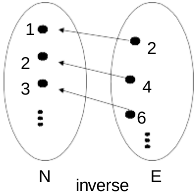
$f^{-1}(y) = (y - 5)/2$



Example

- The inverse function of $f(x) = 2x$ from the set of natural numbers N to the set of non- negative even numbers E is $f^{-1}(x) = 1/2\ x$ from E to N .





Syllabus : DM1

- Sets
- Relations
- Functions
- Recursion Recursion Recursion Recursion

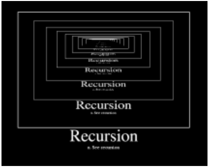
Objective

- Understanding the basic ideas about recursion
- Facility in writing recursive functions

Overview

- Intuitive understanding of recursion
- Formal specification of recursion
- Recursively defined functions

Recursion



- Recursion defines (the recursive) object in terms of itself, i.e. previously defined object of the same class

Recursion

- "In order to understand recursion, you must first understand recursion"
- A recursive function calls itself
- How do you move a stack of 100 boxes?
  - Answer: you move one box, remember where you put it, and then solve the smaller problem: how do you move a stack of 99 boxes?

Recursion

- Solving a "big" problem recursively means to solve one or more smaller versions of the problem, and using those solutions of the smaller problems to solve the "big" problem.
- In particular, solving problems recursively typically means that there are smaller versions of the problem solved in similar ways.

RECURSIVELY DEFINED FUNCTIONS

Recursively defined functions

- A recursive definition of a function defines values of the functions for some inputs in terms of the values of the same function for other inputs.

- How to define a function recursively:

  specify the values of the function for the basic element of the domain

  - e.g., f(0)

- define the value of the function at an element, say x, of the domain by using its value at the predecessor of the element x

  - f(x) in terms of f(x-1)

Example

- The factorial function (denoted as n!) describes the operation of multiplying a number by all the positive integers smaller than it.
- For example, 5! = 5 x 4 x 3 x 2 x 1
- And 9! = 9 x 8 x 7 x 6 x 5 x 4 x 3 x 2 x 1
- And 8! = 8 x 7 x 6 x 5 x 4 x 3 x 2 x 1

- Note: 9! can be written much more concisely:
- 9! = 9 x 8 x 7 x 6 x 5 x 4 x 3 x 2 x 1

  = 9 x 8!

Example

- So: $9! = 8! \times 9$
- Define recursively factorial function: $f(n) = n!$ on N.

  We know that $0! = 1$

- base clause:

  $f(0) = 0! = 1$

- recursive clause:

  $f(9) = f(8) \times 9$

  For all $n \in N$, $f(n) = f(n-1) \times n$

USING RECURSIVE FUNCTIONS

Use Example 1

- Find f(1), f(2) and f(3) if f is defined recursively on the set of natural numbers by:

  $f(n) = f(n-1) + 3$ and $f(0)=5$

- Base clause: $f(0) = 5$
- Recursive clause: $f(n) = f(n-1) + 3$

  $f(1) = f(0) + 3 = 5 + 3 = 8$
  $f(2) = f(1) + 3 = 8 + 3 = 11$
  $f(3) = f(2) + 3 = 11 + 3 = 14$

Use Example 2

- Find f (1), f(2) and f(3) if f is defined recursively on the set of natural numbers by:

  f (n) = 4 x f (n-1) and f(0)=5

- Base clause : $f\,(0) = 5$
- Recursive clause: $f\,(n) = 4 \times f\,(n-1)$

- f(1) = 4 x f(0) = 4 x 5 = 20
- f(2) = 4 x f(1) = 4 x 20 = 80
- f(3) = 4 x f(2) = 4 x 80 = 320

DEFINING RECURSIVE FUNCTIONS

Definition Example 1

- Let f be the function such that f(n) is the sum of the first n positive integers. Give a recursive definition of f(n).

- Answer

  What is the smallest value of n?

  Base clause: f(0) = 0

- How can you define the sum of first n positive integers in terms of the previous sum of (n-1) integers?

Definition Example 1

- How can you define the sum of first n positive integers in terms of the previous sum of (n-1) integers?

$$f(n) = f(n-1) + n$$

- Recursive clause: For all $n \in N$,

$$f(n) = f(n-1) + n$$

- So $f(9) = f(8) + 9$,
  where $f(8)$ is the sum of 1, 2, … 8

Definition Example 2

- Define recursively the function

$$f(n) = 2n + 1 \text{ on } N$$

- Answer:

  base clause: $f(0) = (2*0) + 1 = 1$

Developing the Recursive Clause

- $f(n) = 2n + 1$
- To work out what f(n-1) is, we “plug in” n-1 in the above formula.
- We replace “n” by “n-1” throughout.
- $f(n-1) = 2(n-1) + 1$

  $= 2n - 2 + 1$

  $= 2n - 1$
- We do a little algebra to tidy up the result.

Developing the Recursive Clause

- We wish to express f(n) in terms of f(n-1).
  $f(n) = f(n-1) + B$.
- "B" is some unknown quantity that we need to find.
- On the previous slide, we defined f(n-1) as
  $f(n-1) = 2n - 1$
- So we substitute "2n -1" for "f(n-1)" in the first equation above, giving
  $\mathbf{f(n) = 2n - 1 + B}$

Developing the Recursive Clause

- $f(n) = 2n - 1 + B$
- We know from the original specification that $f(n) = 2n + 1$
- So we substitute "2n + 1" for "f(n)" in the first equation on this slide, giving

  $2n + 1 = 2n - 1 + B$
- Which is the same as

  $2n - 1 + B = 2n + 1$

- $2n - 1 + B = 2n + 1$
- We are trying to find out what B is, so we want an equation with B on its own on the LHS.

$2n -1 + B - 2n = 2n + 1 - 2n$

$= -1 + B = 1$

$= -1 + B + 1 = 1 + 1$

$B = 2$

$f(n) = f(n-1) + B$

$f(n) = f(n-1) + 2$

This is our recursive definition for f(n)

Definition Example 3

- Define recursively: the function $f(n) = 2^n$ on N

- What is the smallest value of n?

  Base clause: $f(0) = 2^0 = 1$

- How can you define a power of 2 in terms of previous power of 2?

  $f(n+1)= 2^{n+1}$

  $f(n+1) = 2^n \times 2$

- Recursive clause: For all $n \in N$, $f(n+1) = f(n) \times 2$

Also the end of part one of Discrete Maths

THE END