# CS1112
# Functions I

**Lecturer:**

Professor Barry O'Sullivan

Office: 2-65, Western Gateway Building

email: *b.osullivan@cs.ucc.ie*

http://osullivan.ucc.ie/teaching/cs1112/

# Functions

function: a relationship between two sets,

translating an input element to an output

functions in computer science

# What are Functions?

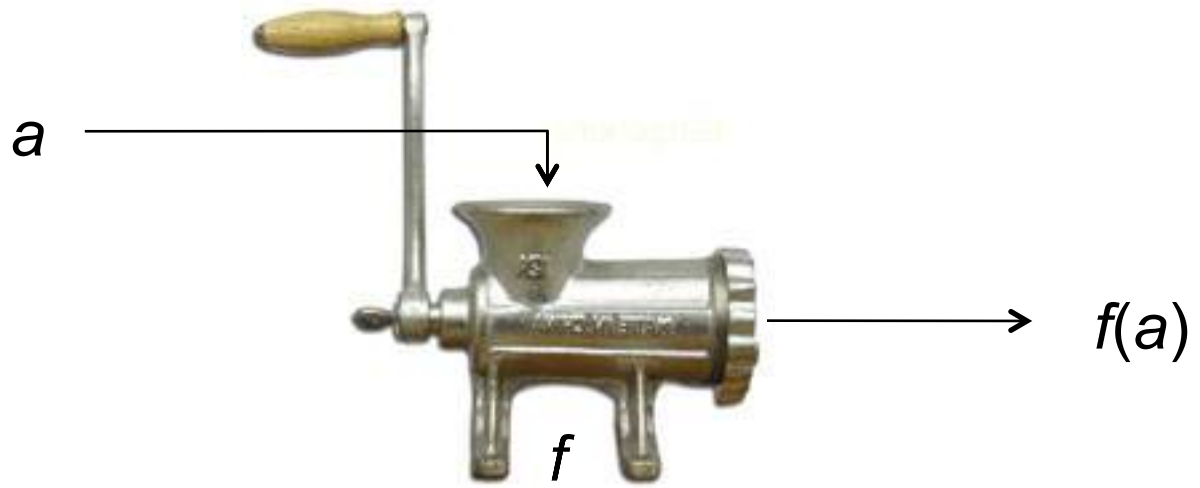
We have seen functions many times in school-level mathematics

- sin, cos, tan
- $f(x) = x^2$
- $f(x) = 5x + 2$
- etc.

The concept is used throughout mathematics, engineering, physics, statistics, economics, etc., and in particular, it is used throughout computer science.

We need to understand exactly what a function is, and understand the different properties it could have.

Think of a function as a machine – we feed it some input, and we get some output in return




In some sense, we don't need to know what goes on inside the machine – all we care about is what output we get for different inputs.

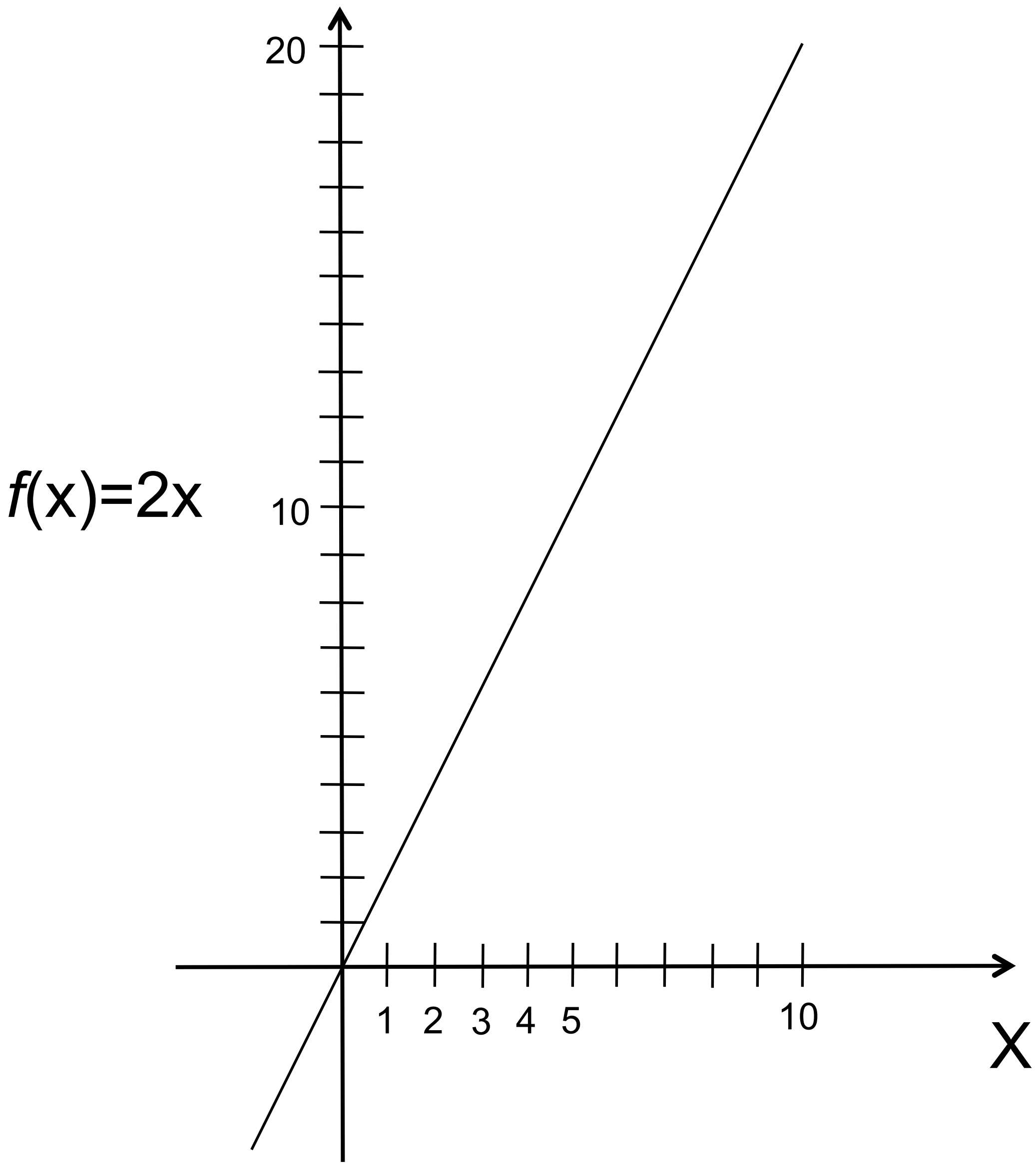
For standard functions in maths, we can sketch them using a graph

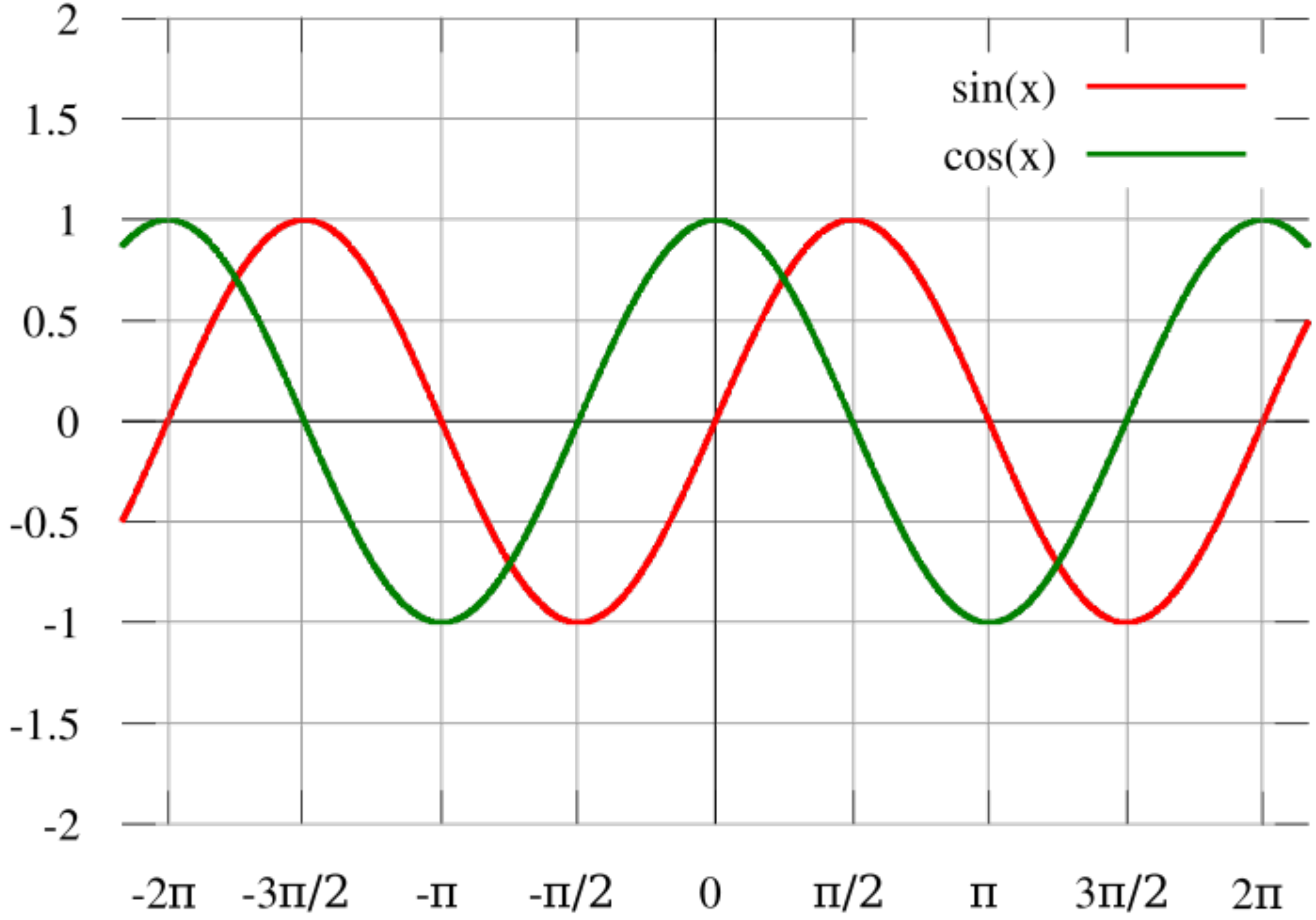
input

$f(x)=2x$

function name

how the output relates to the input




2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
sin(x)
cos(x)
-2π
-3π/2
-π
-π/2
0
π/2
π
3π/2
2π

# General functions

Functions are more general than those seen in maths; e.g.

- given an ID number as input, a function returns the name of the student with that ID
- given a country in the United Nations, a function returns the name of the capital city
- given the name of a month, and a year, in *python* a function "cal_days_in_month" returns the number of days in that month
- given the text of an email, a function "encrypt" returns an encrypted version of the text

# So what is a function?

To start with, we only consider functions between two sets.

Let $A$ and $B$ be two non-empty sets. A function $f$ from $A$ to $B$ specifies for each element of $A$ exactly one element of $B$.

We write $f : A \to B$, and if $f$ specifies $b$ for $a$, we write $f(a)=b$.

If $f : A \to B$, then $A$ is the domain, and B is the codomain.

# What properties must a function have?

Every element of the domain must be matched with exactly one element of the codomain (and not matched with anything else)

- every domain element has a partner in the codomain
- no domain element has two or more partners

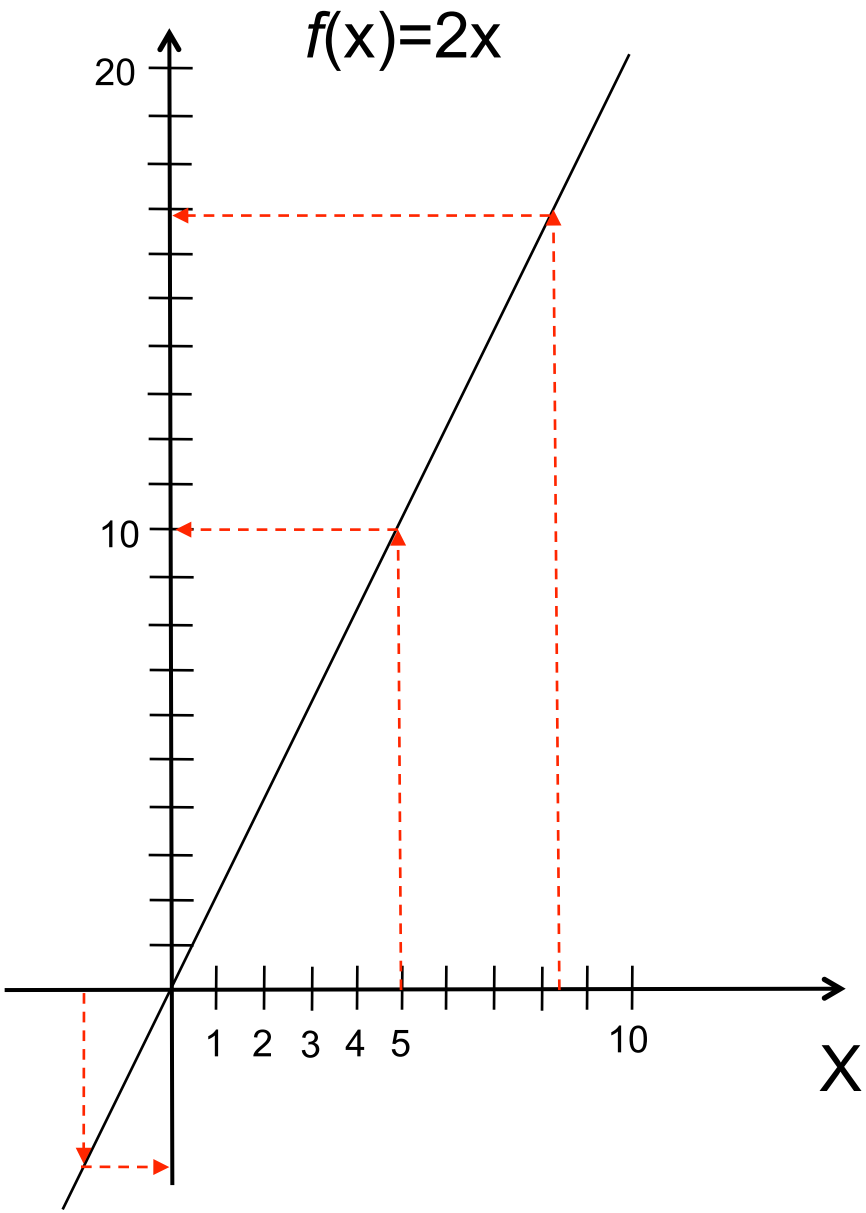
If I give you a valid piece of input, your function must give me a single piece of output  from the codomain, and it must be the same output each time I give you the same input.

domain = R (the set of real numbers)

codomain = R

$f : R \rightarrow R$

each element of the domain has exactly one partner in the codomain



If we can describe the mapping in simple terms, we can write it in the function specification.

For example, $f(x) = 2x$ is $\quad f : R \rightarrow R : x \mapsto 2x$

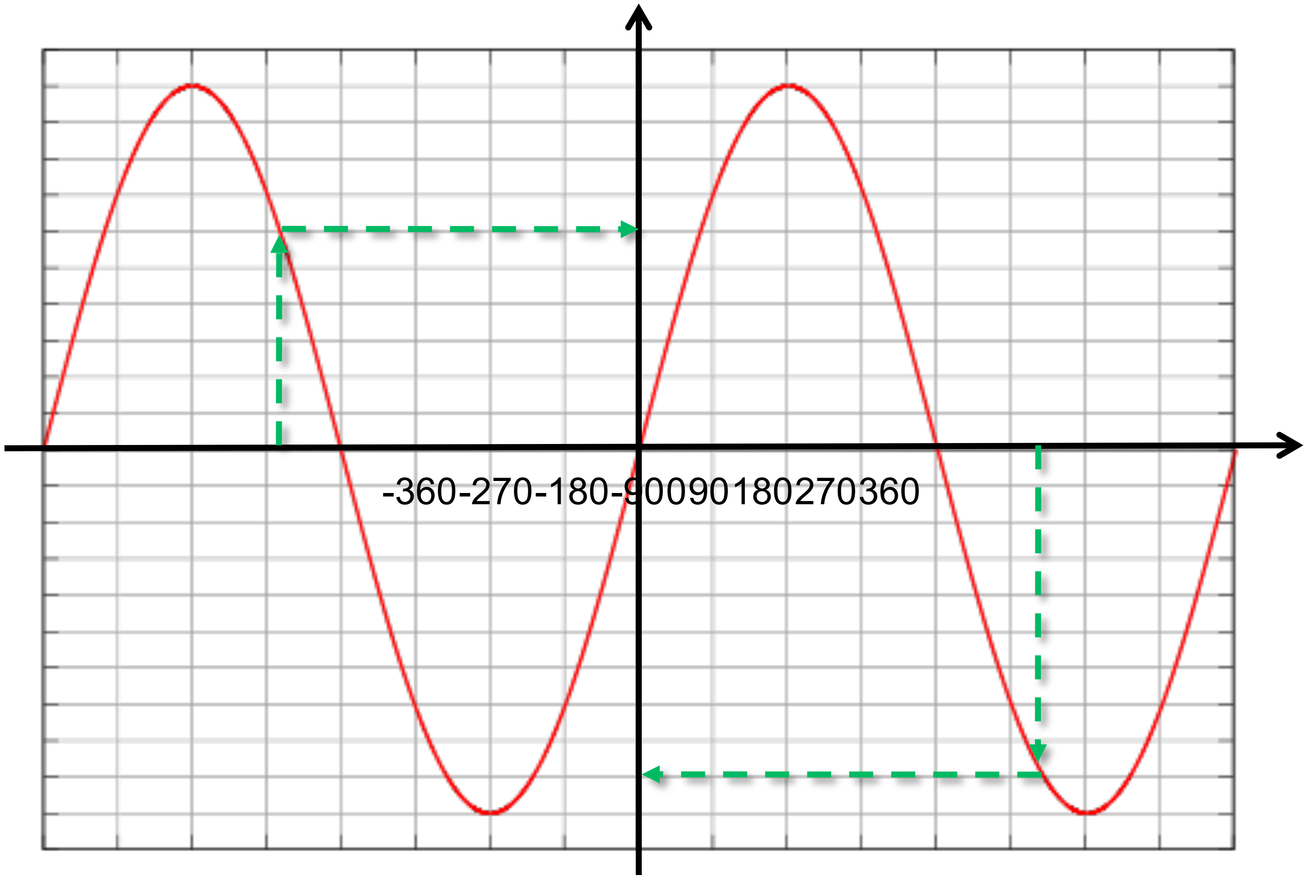
- "$\mapsto$" is called the maplet, and we use it to specify what a general element $x$ is mapped to
- the right hand side (e.g. "2x" above) can have different cases

Example: $g$ is a function that maps students in UCC to {true,false} depending on whether or not they are doing CS.

$$g : \text{Students} \rightarrow \{\text{true,false}\} : s \mapsto \begin{cases} \text{true, if } s \text{ does CS} \\ \text{false otherwise} \end{cases}$$

domain = R

codomain = R




each element of the domain has exactly one partner in the codomain

$$f : R \to R : x \mapsto sin(x)$$

Example: represent the function "adds 1 to" which acts on integers, and simply adds 1 to the number you use as input.

Example: represent the function "is even" which maps integers to {true,false}, and return true if the input is even, and false if the input is odd.

# Representing functions

- There are many different ways we can represent a function, which are useful for functions which do not map numbers onto other numbers. E.g.

| | Cork | Kerry | Waterford | Limerick | Tipperary | Clare |
|---|---|---|---|---|---|---|
| mallow | X | | | | | |
| killarney | | X | | | | |
| dungarvan | | | X | | | |
| limerick | | | | X | | |
| ennis | | | | | | X |
| midleton | X | | | | | |
| tralee | | X | | | | |
| bantry | X | | | | | |

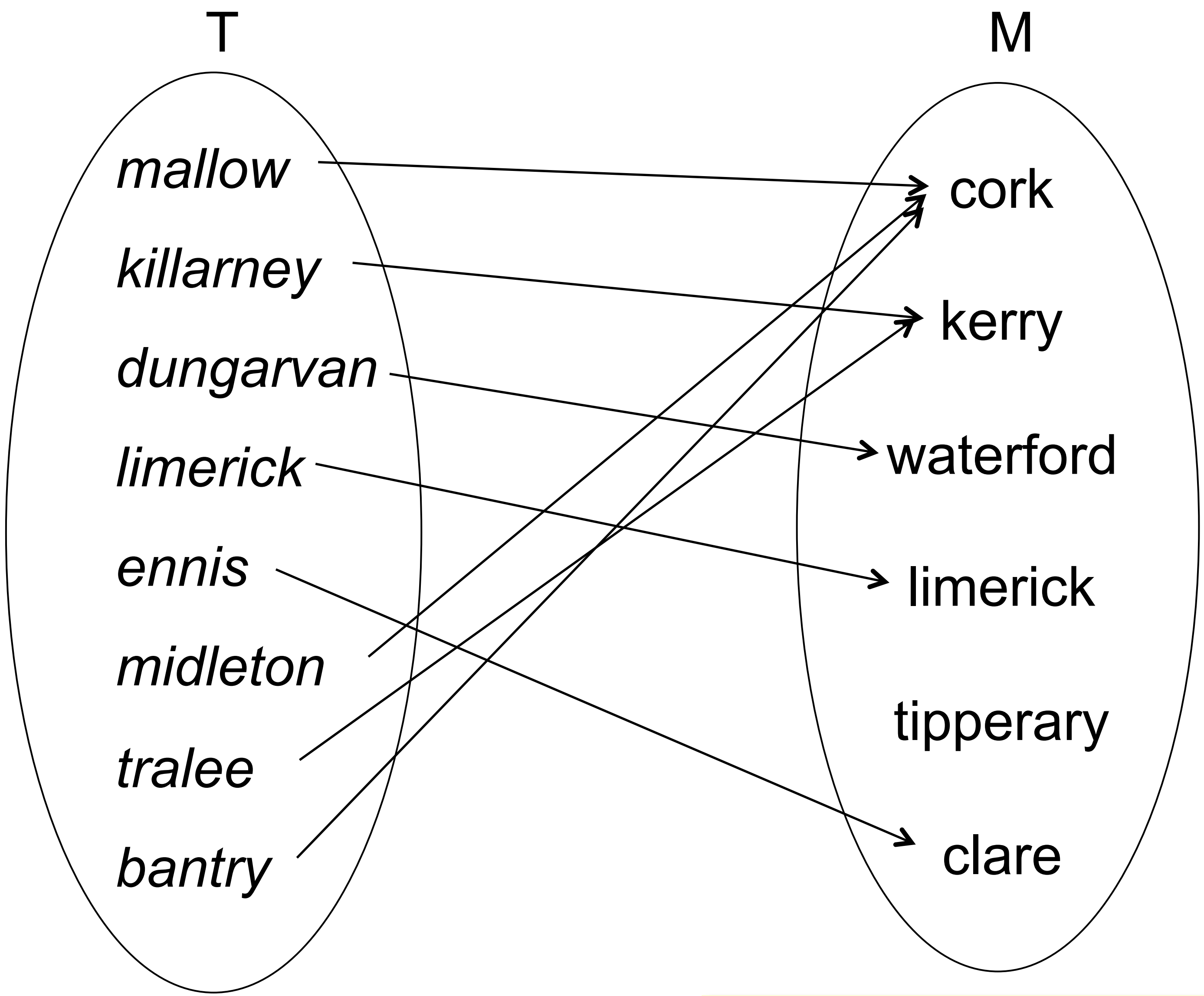
... using a matrix

*T= {mallow,killarney,dungarvan,limerick,ennis,middleton,tralee, bantry}*
*M = {Cork, Kerry,Waterford,Limerick,Tipperary,Clare}*
$f : T \to M$

# Representing functions

| T | M |
|---|---|
| *mallow* | Cork |
| *killarney* | Kerry |
| *dungarvan* | Waterford |
| *limerick* | Limerick |
| *ennis* | Clare |
| *midleton* | Cork |
| *tralee* | Kerry |
| *bantry* | Cork |

... using a lookup table




... using an arrow diagram

We will use arrow diagrams for sketching examples

{ (mallow,Cork), (killarney,Kerry) , (dungarvan,Waterford), (limerick,Limerick), (ennis, Clare), (midleton,Cork), (tralee, Kerry), (bantry,Cork)}

A set of ordered pairs over two sets is a subset of the Cartesian product

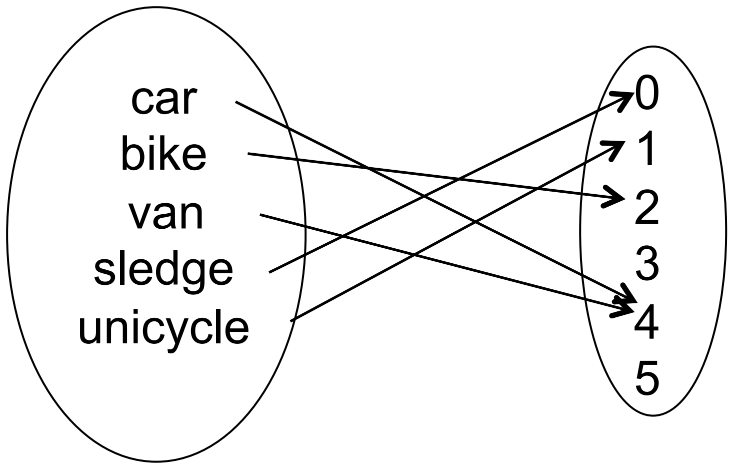
... using a set of ordered pairs

We will use ordered pairs as our standard formal representation

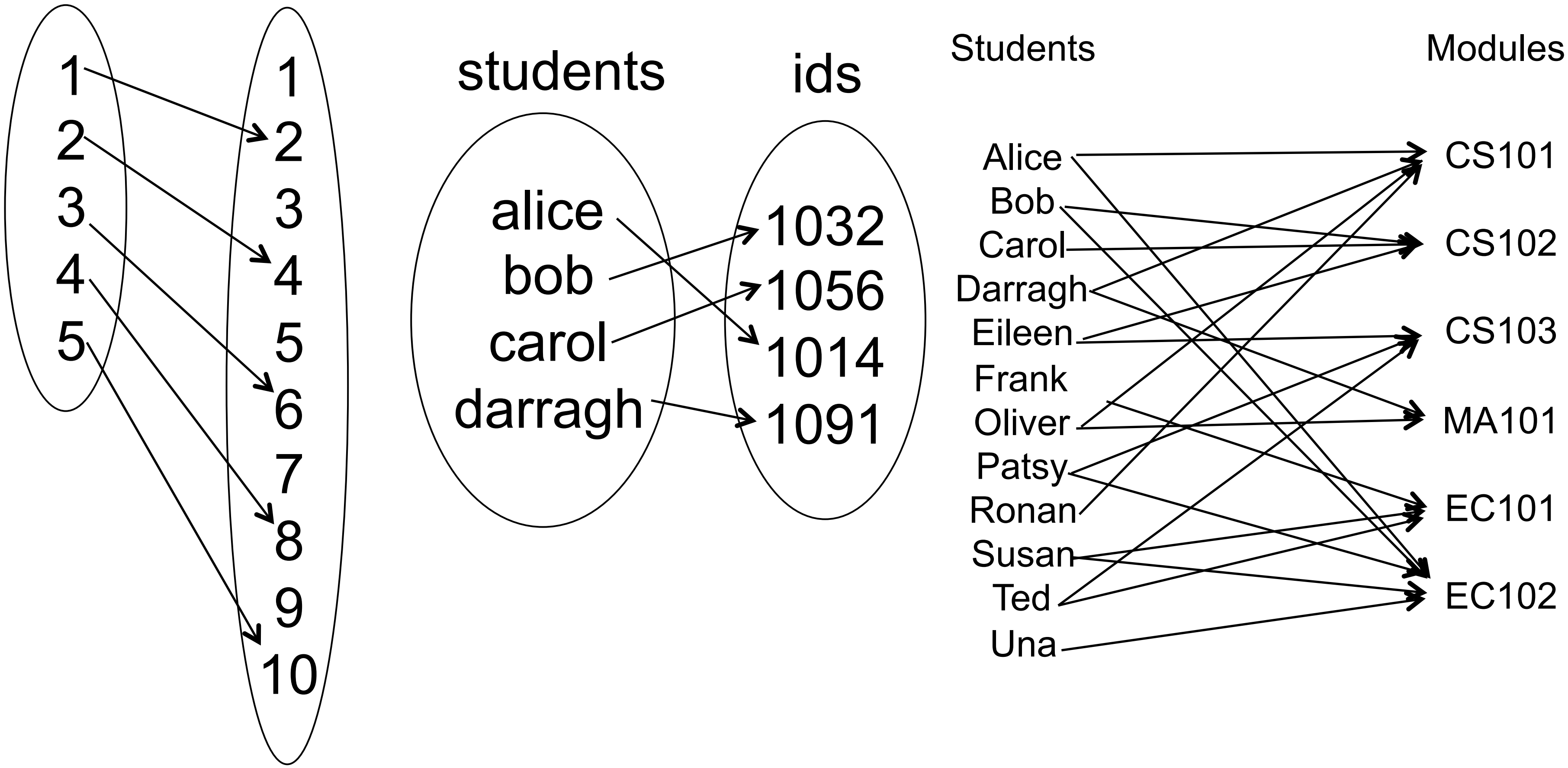
# Exercise

Consider the function sketched below as an arrow diagram.

- What are the domain and codomain of the function?



# Are these functions?



Example: are the following relationships functions? If so, specify the function formally; if not, state why not.

1. From the integers to the integers, which when given some input, returns the number which is exactly half the input.

2. From the real numbers to the real numbers, which when given some input, returns the number which is exactly half the input.

3. From the real numbers to the integers, which when given some input, returns the integer obtained by discarding the part after the decimal point

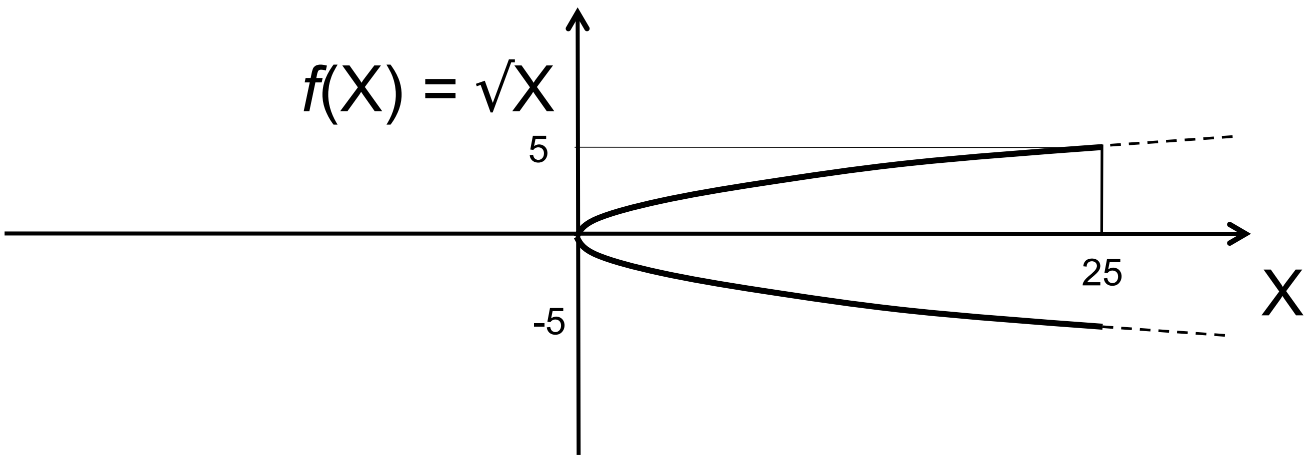
Note: square-root is **not** a function from R to R.

- There is no corresponding element for -1, for example.

Square root is **not** a function from the non-negative R to R.

- There are two corresponding elements for 4, for example – root(4) could be +2 or -2

Square root **is** a function from non-negative R to non-negative R - each element has exactly one corresponding element.



# Extended example: project groups

U={Alice, Bob, Carol, Darragh, Eileen, Frank, Oliver, Patsy, Ronan, Susan, Ted,Una}
group1 = {Alice, Bob, Carol}
group2 = {Darragh Eileen, Frank}
group3 = {Oliver, Patsy, Ronan}
group4 = {Susan, Ted, Una}

The function *group* assigns a group number to each student.
The function *leader* assigns a leader to each group.

# functions in algorithms and programming languages

In an algorithm or computer program, a function is a separate piece of program code which takes input of a particular type, and produces output of a particular type.

In most languages, we must specify the types of input and output (i.e. domain and co-domain) when we write the function.

E.g.

domain

```
int floor(double z);
```

codomain

function name

is the Java header for a function called *floor*, which says the input data must be of type `double` (a real number), and the output will be of type `int` (an integer)

# Some standard functions

*floor*: $\mathbb{R} \rightarrow \mathbb{Z} : x \mapsto$ the largest integer $\leq x$

*ceil*: $\mathbb{R} \rightarrow \mathbb{Z} : x \mapsto$ the smallest integer $\geq x$

```
int ceil(double x)
```

*ceil* takes a real number (or *double*) as input, and gives an integer (or *int*) as output.
For any given real number input $x$, the output will be the smallest integer that is not smaller than $x$

*abs:* $\mathbb{R} \rightarrow \mathbb{R}^+ : x \mapsto$ the greater of $x$ and $-x$

# standard function concepts

A function $f$: R→ R is linear if $f(x)$ can be written $f(x) = ax+b$ for some fixed integers $a$ and $b$.

A function $f$: R → R is polynomial if $f(x)$ can be written
$f(x) = a_1x^n + a_2x^{(n-1)} + ... a_{n-1}x + a_n$

for some fixed integers $n$ and $a_1, a_2, ..., a_n$.

A function $f$ : R →R is exponential if $f(x)$ contains a term $...a^x$ for some fixed integer $a$.

These concepts will be used heavily in later years, when studying how long algorithms take to run.

# Next lecture ...

Properties of functions

Combining functions