



COMP9020

Foundations of Computer Science

Lecture 7: Functions

Administrivia

- First assignment available today – due **October 20th, 12 noon**
- Next batch of formatif tasks will be made available today - generally fewer tasks, Wednesday deadlines
- Feedback:
 - Amount of content

Applications of Functions and Big-O notation

- Functions, methods, procedures in programming
- Computer programs “are” functions
- Graphical transformations
- Algorithmic analysis

Outline

Functions Recap

Functional Composition

Inverse Functions

Matrices

Introduction to Big-O Notation

Feedback

Outline

Functions Recap

Functional Composition

Inverse Functions

Matrices

Introduction to Big-O Notation

Feedback

Properties of Binary Relations $R \subseteq S \times T$

A binary relation $R \subseteq S \times T$ is:

Definition

(Fun)	functional	For all $s \in S$ there is at most one $t \in T$ such that $(s, t) \in R$
(Tot)	total	For all $s \in S$ there is at least one $t \in T$ such that $(s, t) \in R$
(Inj)	injective	For all $t \in T$ there is at most one $s \in S$ such that $(s, t) \in R$
(Sur)	surjective	For all $t \in T$ there is at least one $s \in S$ such that $(s, t) \in R$
(Bij)	bijective	Injective and surjective

Functions

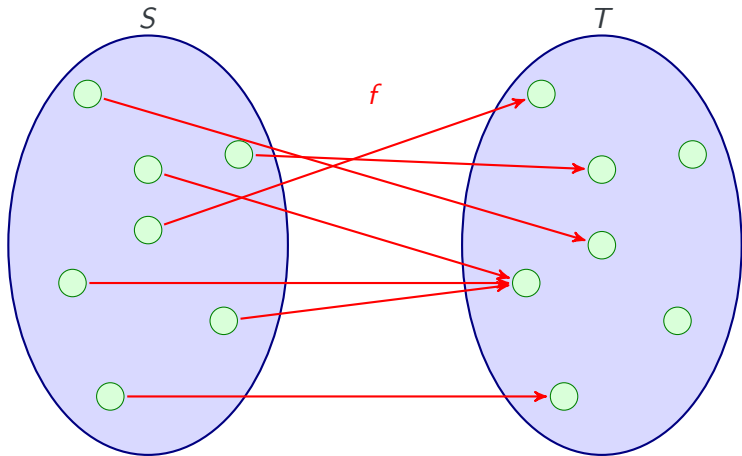
Definition

A **function**, $f : S \rightarrow T$, is a binary relation $f \subseteq S \times T$ that satisfies (Fun) and (Tot). That is, for all $s \in S$ there is *exactly one* $t \in T$ such that $(s, t) \in f$.

We write $f(s)$ for the unique element related to s .

We write T^S for the set of all functions from S to T .

Graphical representation



Functions

$f : S \longrightarrow T$ describes pairing of the sets: it means that f assigns to every element $s \in S$ a unique element $t \in T$. To emphasise where a specific element is sent, we can write $f : x \mapsto y$, which means the same as $f(x) = y$

		Symbol	
S	domain of f	$\text{Dom}(f)$	(inputs)
T	co-domain of f	$\text{Codom}(f)$	(<i>possible</i> outputs)
$f(S)$	image of f	$\text{Im}(f)$	(<i>actual</i> outputs)
$= \{ f(x) : x \in \text{Dom}(f) \}$			

Example

Example

The **identity** function on S

$$\text{Id}_S(x) = x, x \in S$$

- $\text{Dom}(\text{Id}_S) = S$
- $\text{Codom}(\text{Id}_S) = S$
- $\text{Im}(\text{Id}_S) = S$

Important!

The domain and co-domain are critical aspects of a function's definition.

$$f : \mathbb{N} \rightarrow \mathbb{Z} \quad \text{given by} \quad f(x) = x^2$$

and

$$g : \mathbb{N} \rightarrow \mathbb{N} \quad \text{given by} \quad g(x) = x^2$$

are different functions even though they have the same behaviour!

Injective functions

Function $f : S \rightarrow T$ is called an **injection** or **1-1 (one-to-one)** if it satisfies (Inj)

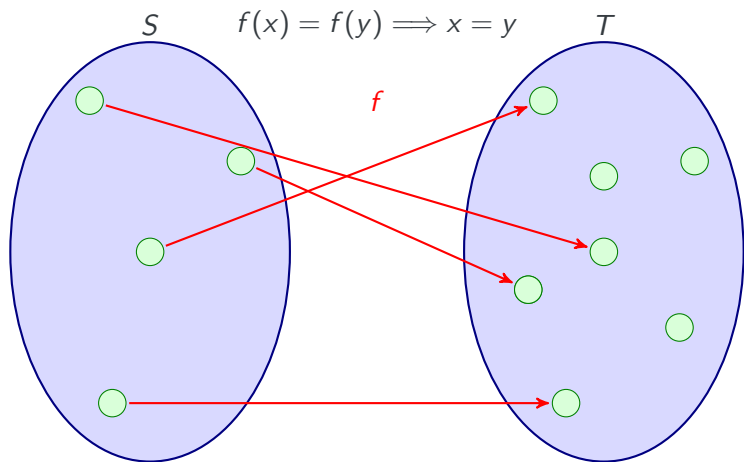
Examples (of functions that are injective)

- $f : \mathbb{N} \rightarrow \mathbb{N}$ with $f(x) \mapsto x$
- set complement (for a fixed universe)

Examples (of functions that are not injective)

- absolute value, floor, ceiling
- length of a word

Graphical representation: Injective



Surjective functions

Function $f : S \rightarrow T$ is called a **surjection** or **onto** if it satisfies (Sur). That is, if

$$\text{Im}(f) = \text{Codom}(f)$$

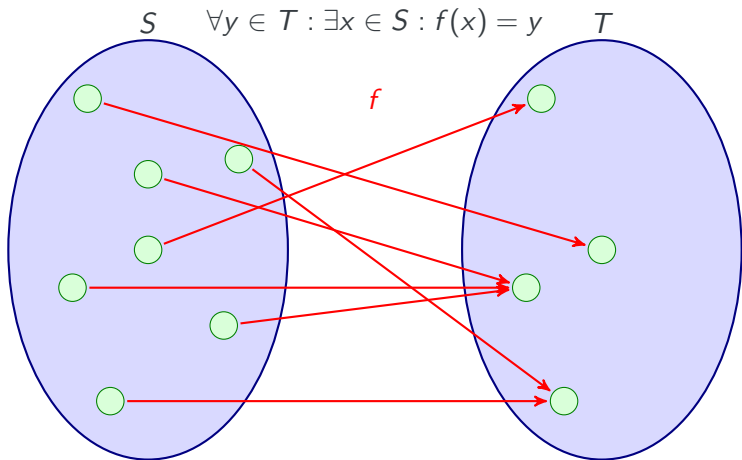
Examples (of functions that are surjective)

- $f : \mathbb{N} \rightarrow \mathbb{N}$ with $f(x) \mapsto x$
- Floor, ceiling

Examples (of functions that are not surjective)

- $f : \mathbb{N} \rightarrow \mathbb{N}$ with $f(x) \mapsto x^2$
- $f : \{a, \dots, e\}^* \rightarrow \{a, \dots, e\}^*$ with $f(w) \mapsto awe$

Graphical representation: Surjective



Functions on finite sets

NB

For a **finite** set S and $f : S \longrightarrow S$ the properties

① *surjective*, and

② *injective*

are equivalent.

Outline

Functions Recap

Functional Composition

Inverse Functions

Matrices

Introduction to Big-O Notation

Feedback

Composition of functions

Question

*If $f : S \rightarrow T$ and $g : T \rightarrow U$ are functions, then $f;g$ is a relation.
When is it a function?*

Composition of Functions

Definition

If $f : S \rightarrow T$ and $g : T \rightarrow U$ then the **composition of f and g** , written $g \circ f$, is the function given by

$$(g \circ f)(x) = g(f(x)).$$

That is, $g \circ f = f; g$.

Facts

- Composition is associative

$$h \circ (g \circ f) = (h \circ g) \circ f$$

- For $g : S \rightarrow T$

$$g \circ \text{Id}_S = g \quad \text{and} \quad \text{Id}_T \circ g = g.$$

Iteration of Functions

If a function maps a set into itself, i.e. when $\text{Dom}(f) = \text{Codom}(f)$, the function can be composed with itself — **iterated**

$$f \circ f, f \circ f \circ f, \dots, \quad \text{also written } f^2, f^3, \dots$$

Exercises

Exercises

Let $f, g : \mathbb{Z} \rightarrow \mathbb{Z}$ be given by $f(n) = n^2 + 3$ and $g(n) = 5n - 11$.
What is:

- $f \circ g(n) =$
- $g \circ f(n) =$
- $g^2(n) =$

Outline

Functions Recap

Functional Composition

Inverse Functions

Matrices

Introduction to Big-O Notation

Feedback

Converse of a function

Question

f^{\leftarrow} is a relation; when is it a function?

Inverse Functions

Definition

If f^{\leftarrow} is a function then it is called the **inverse function**; denoted f^{-1} .

NB

f^{-1} *only exists if f is a bijection.*

f^{\leftarrow} *always exists.*

f^{-1} is the procedure of “undoing” f .

Properties of the inverse

Fact

If $f : S \rightarrow T$ and f^{-1} exists then:

$$f^{-1} \circ f = Id_S \quad \text{and} \quad f \circ f^{-1} = Id_T.$$

Conversely, if $f : S \rightarrow T$ and $g : T \rightarrow S$ and

$$g \circ f = Id_S \quad \text{and} \quad f \circ g = Id_T$$

then f^{-1} exists and is equal to g .

Exercises

Exercises

RW: 1.7.5 f and g are 'shift' functions $\mathbb{N} \rightarrow \mathbb{N}$ defined by $f(n) = n + 1$, and $g(n) = \max(0, n - 1)$

(c) Is f injective? surjective?

(d) Is g injective? surjective?

(e) Do f and g commute, i.e. $\forall n ((f \circ g)(n) = (g \circ f)(n))$?

Exercises

Exercises

RW: 1.7.6 $\Sigma = \{a, b, c\}$

(c) Is $\text{length} : \Sigma^* \rightarrow \mathbb{N}$ surjective?

(d) $\text{length}^{\leftarrow}(2) \stackrel{?}{=}$

RW: 1.7.12 Verify that $f : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R} \times \mathbb{R}$ defined by $f(x, y) = (x + y, x - y)$ is invertible.

Outline

Functions Recap

Functional Composition

Inverse Functions

Matrices

Introduction to Big-O Notation

Feedback

Matrices

An $m \times n$ **matrix** is a rectangular array with m horizontal rows and n vertical columns.

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

NB

Matrices are important objects in Computer Science, e.g. for

- *optimisation*
- *graphics and computer vision*
- *cryptography*
- *information retrieval and web search*
- *machine learning*

Matrix Motivation

Solving linear equations:

$$5x = 15$$

$$5x + 3y = 15$$

$$4x - 2y = 12$$

$$A = \begin{pmatrix} 5 & 3 \\ 4 & -2 \end{pmatrix} \quad \mathbf{x} = \begin{pmatrix} x \\ y \end{pmatrix} \quad \mathbf{b} = \begin{pmatrix} 15 \\ 12 \end{pmatrix}$$

$$A\mathbf{x} = \mathbf{b}$$

$$x' = 5x + 3y \quad x'' = 2x' + y'$$

$$y' = 4x - 2y \quad y'' = 3x' + 3y'$$

$$A = \begin{pmatrix} 5 & 3 \\ 4 & -2 \end{pmatrix} \quad \mathbf{x} = \begin{pmatrix} x \\ y \end{pmatrix} \quad \mathbf{x}' = \begin{pmatrix} x' \\ y' \end{pmatrix}$$

$$B = \begin{pmatrix} 2 & 1 \\ 3 & 3 \end{pmatrix} \quad \mathbf{x}'' = \begin{pmatrix} x'' \\ y'' \end{pmatrix}$$

Basic Matrix Operations

The **transpose** \mathbf{A}^T of an $m \times n$ matrix $\mathbf{A} = [a_{ij}]$ is the $n \times m$ matrix whose entry in the i th row and j th column is a_{ji} .

Example

$$\mathbf{A} = \begin{bmatrix} 2 & -1 & 0 & 4 \\ 3 & 2 & -1 & 2 \\ 4 & 0 & 1 & 3 \end{bmatrix}$$

$$\mathbf{A}^T = \begin{bmatrix} 2 & 3 & 4 \\ -1 & 2 & 0 \\ 0 & -1 & 1 \\ 4 & 2 & 3 \end{bmatrix}$$

NB

A matrix \mathbf{M} is called symmetric if $\mathbf{M}^T = \mathbf{M}$

Matrix Sum

The **sum** of two $m \times n$ matrices $\mathbf{A} = [a_{ij}]$ and $\mathbf{B} = [b_{ij}]$ is the $m \times n$ matrix whose entry in the i th row and j th column is $a_{ij} + b_{ij}$.

Example

$$\mathbf{A} = \begin{bmatrix} 2 & -1 & 0 & 4 \\ 3 & 2 & -1 & 2 \\ 4 & 0 & 1 & 3 \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 1 & 0 & 5 & 3 \\ 2 & 3 & -2 & 1 \\ 4 & -2 & 0 & 2 \end{bmatrix}$$

$$\mathbf{A} + \mathbf{B} = \begin{bmatrix} 3 & -1 & 5 & 7 \\ 5 & 5 & -3 & 3 \\ 8 & -2 & 1 & 5 \end{bmatrix}$$

Fact

$$\mathbf{A} + \mathbf{B} = \mathbf{B} + \mathbf{A} \quad \text{and} \quad (\mathbf{A} + \mathbf{B}) + \mathbf{C} = \mathbf{A} + (\mathbf{B} + \mathbf{C})$$

Scalar Product

Given $m \times n$ matrix $\mathbf{A} = [a_{ij}]$ and $c \in \mathbb{R}$, the **scalar product** $c\mathbf{A}$ is the $m \times n$ matrix whose entry in the i th row and j th column is $c \cdot a_{ij}$.

Example

$$\mathbf{A} = \begin{bmatrix} 2 & -1 & 0 & 4 \\ 3 & 2 & -1 & 2 \\ 4 & 0 & 1 & 3 \end{bmatrix}$$

$$2\mathbf{A} = \begin{bmatrix} 4 & -2 & 0 & 8 \\ 6 & 4 & -2 & 4 \\ 8 & 0 & 2 & 6 \end{bmatrix}$$

Matrix Product

The **product** of an $m \times n$ matrix $\mathbf{A} = [a_{ij}]$ and an $n \times p$ matrix $\mathbf{B} = [b_{jk}]$ is the $m \times p$ matrix $\mathbf{C} = [c_{ik}]$ defined by

$$c_{ik} = \sum_{j=1}^n a_{ij} b_{jk} \quad \text{for } 1 \leq i \leq m \text{ and } 1 \leq k \leq p$$

Example

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \cdot \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \end{bmatrix}$$

NB

The **rows** of \mathbf{A} must have the same number of entries as the **columns** of \mathbf{B} .

The product of a $1 \times n$ matrix and an $n \times 1$ matrix is usually called the **inner product** of two **n-dimensional vectors**.

Example

Example

Consider

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 2 & -1 \\ -6 & 3 \end{bmatrix}$$

Calculate \mathbf{AB} , \mathbf{BA}

$$\mathbf{AB} = \begin{bmatrix} -10 & 5 \\ -20 & 10 \end{bmatrix} \quad \mathbf{BA} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

NB

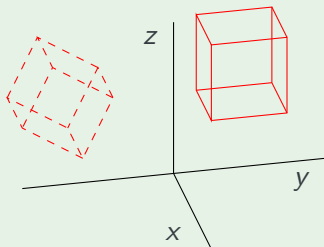
In general, $\mathbf{A} \cdot \mathbf{B} \neq \mathbf{B} \cdot \mathbf{A}$

Example: Computer Graphics

Example

Rotating an object w.r.t. the x axis by degree α :

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \alpha & -\sin \alpha \\ 0 & \sin \alpha & \cos \alpha \end{bmatrix} \cdot \begin{bmatrix} 5 & 5 & 7 & 7 & 5 & 7 & 5 & 7 \\ 1 & 1 & 1 & 1 & 3 & 3 & 3 & 3 \\ 9 & 7 & 7 & 9 & 7 & 7 & 9 & 9 \end{bmatrix}$$



Outline

Functions Recap

Functional Composition

Inverse Functions

Matrices

Introduction to Big-O Notation

Feedback

Motivation

Want to compare functions, particularly functions from \mathbb{N} to \mathbb{R}

Options:

- Equality: $f(n) = g(n)$ for all n
- (Pointwise) comparison: $f(n) \leq g(n)$ for all n
- (Almost all) comparison: $f(n) \leq g(n)$ for all but finitely many n
- Asymptotic growth: $\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)}$

Motivating example: Algorithmic analysis

Example

Want to compare algorithms – particularly ones that can solve *arbitrarily large* instances.

We would like to be able to talk about the resources (running time, memory, energy consumption) required by a program/algorithm as a function $f(n)$ of some parameter n (e.g. the size) of its input.

e.g. How long does a given sorting algorithm take to run on a list of n elements?

Motivating example: Algorithmic analysis

Issues

- The exact resources required for an algorithm are difficult to pin down. Heavily dependent on:
 - Environment the program is run in (hardware, choice of language, external factors, etc)
 - Choice of inputs used
- Cost functions can be complex, e.g.

$$2n \log(n) + (n - 100) \log(n)^2 + \frac{1}{2^n} \log(\log(n))$$

Need to identify the “important” aspects of the function.

Solution

Look at the **asymptotic growth**: how do the costs **scale** as n gets large?

“Big-O” Asymptotic Upper Bounds

Definition

Let $f, g : \mathbb{N} \rightarrow \mathbb{R}_{\geq 0}$. We say that g is *asymptotically less than* f (or: **f is an upper bound of g**) if there exists $n_0 \in \mathbb{N}$ and a real constant $c > 0$ such that for all $n \geq n_0$,

$$g(n) \leq c \cdot f(n)$$

Write $O(f(n))$ for the class of all functions g that are asymptotically less than f .

Example

$$g(n) = 3n + 1 \implies g(n) \leq 4n, \text{ for all } n \geq 1$$

Therefore, $3n + 1 \in O(n)$

Example

$$\frac{1}{10}n^2 \in O(n^2) \quad 10n \log n \in O(n \log n) \quad O(n \log n) \subsetneq O(n^2)$$

The traditional notation has been

$$g(n) = O(f(n))$$

instead of $g(n) \in O(f(n))$.

It allows one to use $O(f(n))$ or similar expressions as part of an equation; of course these 'equations' express only an approximate equality. Thus,

$$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + O(n)$$

means

"There exists a function $f(n) \in O(n)$ such that $T(n) = 2T(\frac{n}{2}) + f(n)$."

Alternative definition

Fact

$$f(n) \in O(g(n)) \quad \text{if and only if} \quad \lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} < \infty.$$

Properties

Fact

Suppose $f(n) \in O(g(n))$, $g(n) \in O(h(n))$ and $j(n) \in O(k(n))$.

Then:

- $f(n) \in O(h(n))$
- $f(n) + j(n) \in O(g(n) + k(n))$
- $f(n) \cdot j(n) \in O(g(n) \cdot k(n))$

Examples

Examples

$$5n^2 + 3n + 2 \in O(n^2)$$

$$n^3 + 2^{100}n^2 + 2n + 2^{2^{100}} \in O(n^3)$$

Generally, for constants $a_k \dots a_0$,

$$a_k n^k + a_{k-1} n^{k-1} + \dots + a_0 \in O(n^k)$$

“Big-Omega” Asymptotic Lower Bounds

Definition

Let $f, g : \mathbb{N} \rightarrow \mathbb{R}$. We say that g is *asymptotically greater than* f (or: **f is an lower bound of g**) if there exists $n_0 \in \mathbb{N}$ and a real constant $c > 0$ such that for all $n \geq n_0$,

$$g(n) \geq c \cdot f(n)$$

Write $\Omega(f(n))$ for the class of all functions g that are asymptotically greater than f .

Example

$$g(n) = 3n + 1 \implies g(n) \geq 3n, \text{ for all } n \geq 1$$

Therefore, $3n + 1 \in \Omega(n)$

“Big-Theta” Notation

Definition

Two functions f, g have the *same order of growth*, or are **asymptotically equivalent**, if they scale up in the same way: There exists $n_0 \in \mathbb{N}$ and real constants $c > 0, d > 0$ such that for all $n \geq n_0$,

$$c \cdot f(n) \leq g(n) \leq d \cdot f(n)$$

Write $\Theta(f(n))$ for the class of all functions g that have the same order of growth as f .

If $g \in O(f)$ (or $\Omega(f)$) we say that f is an *upper bound* (*lower bound*) on the order of growth of g ; if $g \in \Theta(f)$ we call it a **tight bound**.

Properties

Observe that, somewhat symmetrically

$$g \in \Theta(f) \iff f \in \Theta(g)$$

We obviously have

$$\Theta(f(n)) \subseteq O(f(n)) \quad \text{and} \quad \Theta(f(n)) \subseteq \Omega(f(n)),$$

in fact

$$\Theta(f(n)) = O(f(n)) \cap \Omega(f(n)).$$

At the same time the 'Big-Oh' is *not* a symmetric relation

$$g \in O(f) \not\Rightarrow f \in O(g),$$

but

$$g \in O(f) \Leftrightarrow f \in \Omega(g)$$

Observations

Fact

- For all $k, \epsilon > 0$:

$$O((\log n)^k) \subsetneq O(n^\epsilon) \quad \text{and} \quad O(n^k) \subsetneq O((1 + \epsilon)^n).$$

- All logarithms have the same order, irrespective of base:

$$O(\log_2 n) = O(\log_3 n) = \dots = O(\log_{10} n) = \dots$$

- Exponentials to different bases have different orders:

$$O(r^n) \subsetneq O(s^n) \subsetneq O(t^n) \dots \quad \text{for} \quad r < s < t \dots$$

- Similarly for polynomials

$$O(n^k) \subsetneq O(n^l) \subsetneq O(n^m) \dots \quad \text{for} \quad k < l < m \dots$$

Examples

Examples

Here are some of the most common functions occurring in the analysis of the performance of programs (algorithm complexity), arranged in increasing asymptotic growth:

$1, \log \log n, \log n, \sqrt{n}, \sqrt{n}(\log n), n, n(\log \log n), n \log n, n\sqrt{n}, n^2, n^2 \log n, n^3, n^{12}, 2^{\sqrt{n}}, 1.01^n, 2^n, 3^n, n!, n^n, 2^{n^2}, \dots$

NB

$O(1) \equiv \text{const}$, although technically it could be any function that varies between two constants c and d .

Exercises

Exercises

True or false?

RW: 4.3.5 (a) $2^{n+1} \in O(2^n)$

(b) $(n+1)^2 \in O(n^2)$

(c) $2^{2n} \in O(2^n)$

(d) $(200n)^2 \in O(n^2)$

RW: 4.3.6 (b) $\log(n^{73}) \in O(\log n)$

(c) $\log(n^n) \in O(\log n)$

(d) $(\sqrt{n} + 1)^4 \in O(n^2)$

Outline

Functions Recap

Functional Composition

Inverse Functions

Matrices

Introduction to Big-O Notation

Feedback

Weekly Feedback

I would appreciate any comments/suggestions/requests you have on this week's lectures.



<https://forms.office.com/r/xKKrxYMRn9>