Jun 23 160

90-100	75-89	50-74
A	B	C

20H Per week?

Home work	40 %	← solve problems // each week // submit to TAs // late excepted but 80 pts starting
MT	20 %	← 4th or 5th week
Final	40 %	← during last week

← take home (24 hrs to submit)
open note + open book

No Curve

$$X^n = \underbrace{X \cdot X \cdot \ldots \cdot X}_{n}$$

elements of X^n $(x_1, x_2 \ldots x_n)$

If X and Y are sets (classes), then $Y \subseteq X$ means that Y is a *subset* (subclass) of X, i.e., Y is a set such that all elements of Y belong to X, and X is a *superset* of Y. A subset is *proper* if it does coincide with the whole set.

The *union* $Y \cup X$ of two sets (classes) Y and X is the set (class) that consists of all elements from Y and from X. The union $Y \cup X$ is called disjoint if $Y \cap X = \varnothing$.

The *intersection* $Y \cap X$ of two sets (classes) Y and X is the set (class) that consists of all elements that belong both to Y and to X.

The *union* $\bigcup_{i \in I} X_i$ of sets (classes) X_i is the set (class) that consists of all elements from all sets (classes) X_i, $i \in I$.

The *intersection* $\bigcap_{i \in I} X_i$ of sets (classes) X_i is the set (class) that consists of all elements that belong to each set (class) X_i, $i \in I$.

The *difference* $Y \setminus X$ of two sets (classes) Y and X is the set (class) that consists of all elements that belong to Y but does not belong to X.

If a set (class) X is a subset of a set (class) Y, i.e., $X \subseteq Y$, then the difference $Y \setminus X$ is called the *complement* of the set (class) X in the set (class) Y and is denoted by $C_Y X$.

A fundamental structure of mathematics is *function*. However, functions are special kinds of binary relations between two sets, which are defined below.

A *binary relation* T between sets X and Y, also called *correspondence* from X to Y, is a subset of the direct product $X \times Y$. The set X is called the *domain* of T ($X = \mathrm{Dom}(T)$) and Y is called the *codomain* of T ($Y = \mathrm{Codom}(T)$). The *range* of the relation T is $\mathrm{Rg}(T) = \{ y ; \exists x \in X ((x, y) \in T) \}$. The *domain of definition* also called the *definability domain* of the relation T is $\mathrm{DDom}(T) = \{ x ; \exists y \in Y ((x, y) \in T) \}$. If $(x, y) \in T$, then one says that the elements x and y are in relation T, and one also writes $T(x, y)$.

The image $T(x)$ of an element x from X is the set $\{y; (x, y) \in T\}$ and the coimage $T^{-1}(y)$ of an element y from Y is the set $\{x; (x, y) \in T\}$.

The *graph* of binary relation T between sets of real numbers is the set of points in the two dimensional vector space (a plane), the coordinates of which satisfy this relation.

Binary relations are also called *multivalued functions* (mappings or maps).

Taking binary relations $T \subseteq X \times Y$ and $R \subseteq Y \times Z$, it is possible to build a new binary relation $RT \subseteq X \times Z$ that is called the (*sequential*) *composition* or *superposition* of binary relations T and R and is defined as

$$R \circ T = \{(x, z); x \in X, z \in Z; \text{ where } (x, y) \in T \text{ and } (y, z) \in R \text{ for some } y \in Y\}.$$

A *preorder* (also called *quasiorder*) on a set (class) X is a binary relation Q on X that satisfies the following axioms:

O1. Q is *reflexive*, i.e. xQx for all x from X.

O2. Q is *transitive*, i.e., xQy and yQz imply xQz for all $x, y, z \in X$.

A preorder can be *partial* or *total* when for all $x, y \in X$, we have either xQy or yQx.

A *partial order* is a preorder that satisfies the following additional axiom:

O3. Q is *antisymmetric*, i.e., xQy and yQx imply $x = y$ for all $x, y \in X$.

A *strict* also called *sharp partial order* is a preorder that is not reflexive, is transitive and satisfies the following additional axiom:

O4. Q is *asymmetric*, i.e., only one relation xQy or yQx is true for all $x, y \in X$.

A *linear* or *total order* is a strict partial order that satisfies the following additional axiom:

O5. We have either xQy or yQx for all $x, y \in X$.

satisfies the following additional axiom.

O6. Q is *symmetric*, i.e., xQy implies yQx for all x and y from X.

Set-theoretical symbols

$>$ larger than

$<$ less than

$\geq$ larger than or equal to

$\leq$ less than or equal to

$=$ equal

$\approx$ approximately equal

$\neq$ not equal

$\in$ belongs

$\notin$ does not belong

$\subseteq$ is a subset

$\subset$ is a proper subset

$\not\subset$ is not a proper subset

Traditionally, a *function* (also called a *mapping* or *map* or *total function* or *total mapping* or *everywhere defined function*) f from X to Y is defined as a binary relation between sets X and Y in which there are no elements from X which are corresponded to more than one element from Y and to any element from X, some element from Y is corresponded. At the same time, the function f is also denoted by $f: X \to Y$ or by $f(x)$. In the latter formula, x is a variable and not a definite element from X. The *support*, or *carrier*, of a function f is the closure of the set where $f(x) \neq 0$. Usually the element $f(a)$ is called the *image* of the element a and denotes the value of f on the element a from X. The *coimage* $f^{-1}(y)$ of an element y from Y is the set $\{x; f(x) = y\}$.

However, the traditional definition does not include all kinds of functions and their representations.

There are three basic forms of function representation (definition):

1. (The *set-theoretical*, e.g., *table*, *representation*) A function f is given as a subset R_f of the direct product $X \times Y$ such that the first element if each pair from R_f uniquely defines the second element in this pair, e.g., in a form of a table or of a list of pairs (x, y) where the first element x is taken from X, while the second element y is the image $f(x)$ of the first one. The set R_f is called the *graph* of the function f. When X and Y are sets of points in a geometrical space, e.g., their elements are real numbers, the graph of the function f is called the *geometrical graph* of f.
2. (The *analytic representation*) A function f is described by a formula, i.e., a relevant expression in a mathematical language, e.g., $f(x) = \sin(e^{x + \cos x})$.
3. (The *algorithmic representation*) A function f is given as an algorithm that computes $f(x)$ given x.

$f(x) \equiv a$ means that the function $f(x)$ is equal to a at all points where $f(x)$ is defined.

A function (mapping) f from X to Y is an *injection* if the equality $f(x) = f(y)$ implies the equality $x = y$ for any elements x and y from X, i.e., different elements from X are mapped into different elements from Y.

A function (mapping) f from X to Y is a *bijection* if it is both a projection and injection.

A function (mapping) f from X to Y is an *inclusion* if the equality $f(x) = x$ holds for any element x from X.

Functions

a^x

$\log_a x$

kx^n

$p(x)$

Logical concepts and structures

If P and Q are two statements, then P → Q means that P implies Q and P ↔ Q means that P is equivalent to Q.

Logical operations:

negation is denoted by ¬ or by ~,

conjunction also called logical "and" is denoted by ∧ or by & or by ·,

disjunction also called logical "or" is denoted by ∨,

implication is denoted by → or by ⇒ or by ⊃,

equivalence is denoted by ↔ or by ≡ or by ⇔.

The logical symbol ∀ is called the *universal quantifier* and means "for any".

The logical symbol ∃ is called the *existential quantifier* and means "there exists".

algorithm is a sys of feasable insns for solving some problem

Constructive means that using this description (structure), an automaton (computer) can perform actions prescribed by the algorithm.

The description, e.g., a system of instructions, is a representation of an algorithm.

Why "for solving" and not "of solving"?

Because in a general case, we cannot know if the algorithm actually solves the assigned problem.

Complete or *total algorithm* always solves its problem.

When an algorithm is defined for all its acceptable inputs, then it is called *total*.

In this course, we will study complexity and not any complexity but only **Computational Complexity** of algorithms. However, it is necessary to know that there are other important properties of algorithms.

The most important is **correctness**. If you design an algorithm and if it is not correct, i.e., it does not solve the necessary problem, then you don't do your work properly and who needs employee who don't do their work properly.

Note as one famous physicist said,

Any problem has many easy and … incorrect solutions.

Sometimes it'll be necessary to prove that your algorithm is correct. Sometimes it'll be enough only to demonstrate that your algorithm is correct. In this course, it'll be necessary to prove that your algorithm is correct and I will teach you how to do this.

Another important property of algorithms is **tractability**, which means that it is possible to implement the algorithm and it'll work properly. However, tractability depends on complexity

There are other important properties of algorithms such as

Reliability, Safety, Robustness, Efficiency, Security,

but we don't study them here due to the limitations in time.

Safety of an algorithm means that it does not cause damage to the system that uses it.

Security of an algorithm means that it cannot be damaged by other systems.