CSE 311: Foundations of Computing

Fall 2013

Lecture 10: Functions, Modular arithmetic

announcements

Reading assignment

Modular arithmetic

4.1-4.2, 7th edition

3.4-3.5, 6th edition

Homework 3 due now

Graded Homework 2 and Solutions available

Homework 4 out later today

review: set theory

$x \in A$: "x is an element of A"

$x \notin A$: $\neg (x \in A)$

$$A \subseteq B \equiv \forall x\, (x \in A \rightarrow x \in B)$$

$$A = B \leftrightarrow (A \subseteq B \wedge B \subseteq A)$$

$$A \cup B = \{\, x : (x \in A) \vee (x \in B)\}$$

$$A \cap B = \{\, x : (x \in A) \wedge (x \in B)\}$$

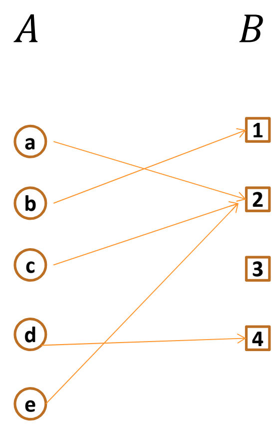
$$\mathcal{P}(A) = \{\, B : B \subseteq A\} \qquad A \times B = \{(a, b) : a \in A, b \in B\}$$

Some applications: Characteristic vectors, private key cryptography

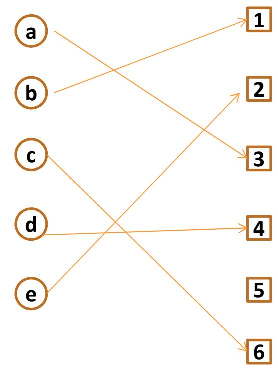
functions review

- A *function* from A to B
  - an assignment of exactly one element of B to each element of A.
  - We write $f : A \rightarrow B$.
  - "Image of a" = $f(a)$
- *Domain* of f: A Codomain of f: B
- *Range* of f = set of all images of elements of A

image, preimage



is this a function? one-to-one? onto?



number theory (and applications to computing)

- Branch of Mathematics with direct relevance to computing
- Many significant applications
  - Cryptography
  - Hashing
  - Security
- Important tool set

modular arithmetic

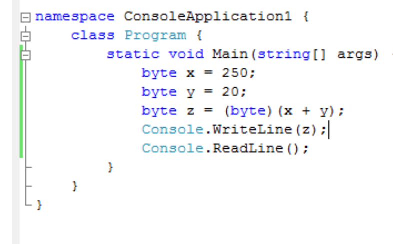
- Arithmetic over a finite domain
- In computing, almost all computations are over a finite domain

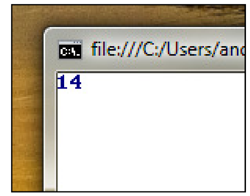
what are the values computed?

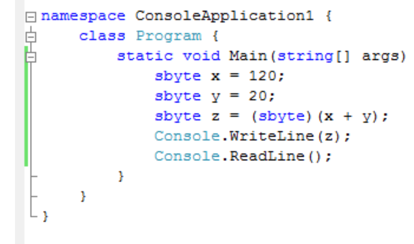
```
public void Test1() {
        byte x = 250;
        byte y = 20;
        byte z = (byte) (x + y);
        Console.WriteLine(z);
}
```

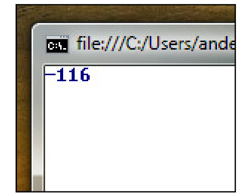
```
public void Test2() {
        sbyte x = 120;
        sbyte y = 20;
        sbyte z = (sbyte) (x + y);
        Console.WriteLine(z);
}
```

what are the values computed?









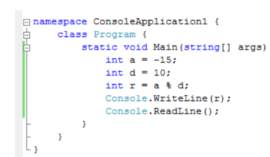
divisibility

Integers a, b, with a ≠ 0, we say that a *divides* b if there is an integer k such that b = ka. The notation a | b denotes "a divides b."

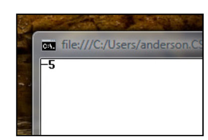
division theorem

Let *a* be an integer and *d* a positive integer. Then there are *unique* integers *q* and *r*, with $0 \le r < d$, such that $a = dq + r$.

$q = a$ **div** d $\qquad$ $r = a$ **mod** d



Note: r ≥ 0 even if a < 0. Not quite the same as `a%d`



arithmetic mod 7

$$a +_7 b = (a + b) \bmod 7$$
$$a \times_7 b = (a \times b) \bmod 7$$

+	0	1	2	3	4	5	6
0							
1							
2							
3							
4							
5							
6							

x	0	1	2	3	4	5	6
0							
1							
2							
3							
4							
5							
6							

modular arithmetic

Let a and b be integers, and m be a positive integer. We say a *is congruent to b modulo m* if m divides a – b. We use the notation a ≡ b (mod m) to indicate that a is congruent to b modulo m.

modular arithmetic

Let a and b be integers, and let m be a positive integer. Then a ≡ b (mod m) if and only if a mod m = b mod m.

modular arithmetic

Let m be a positive integer. If a ≡ b (mod m) and c ≡ d (mod m), then

- a + c ≡ b + d (mod m) and
- ac ≡ bd (mod m)

example

Let n be an integer.
Prove that $n^2 \equiv 0 \pmod 4$ or $n^2 \equiv 1 \pmod 4$

n-bit unsigned integer representation

- Represent integer x as sum of powers of 2:
  If $x = \sum_{i=0}^{n-1} b_i 2^i$ where each $b_i \in \{0,1\}$
  then representation is $b_{n-1}...b_2\, b_1\, b_0$

99 = 64 + 32 + 2 + 1
18 = 16 + 2

- For n = 8:
  99: 0110 0011
  18: 0001 0010

signed integer representation

n-bit signed integers
Suppose $-2^{n-1} < x < 2^{n-1}$
First bit as the sign, n-1 bits for the value

99 = 64 + 32 + 2 + 1
18 = 16 + 2

For n = 8:
99: 0110 0011
-18: 1001 0010

Any problems with this representation?

two's complement representation

n bit signed integers, first bit will still be the sign bit

Suppose $0 \le x < 2^{n-1}$,
x is represented by the binary representation of x
Suppose $0 \le x \le 2^{n-1}$,
$-x$ is represented by the binary representation of $2^n - x$

Key property: Twos complement representation of any number y is equivalent to y mod 2^n so arithmetic works mod 2^n

99 = 64 + 32 + 2 + 1
18 = 16 + 2

For n = 8:
99: 0110 0011
-18: 1110 1110

signed vs two's complement

-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7
1111	1110	1101	1100	1011	1010	1001	0000	0001	0010	0011	0100	0101	0110	0111

Signed

-8	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7
1000	1001	1010	1011	1100	1101	1110	1111	0000	0001	0010	0011	0100	0101	0110	0111

Two's complement

two's complement representation

- For $0 < x \leq 2^{n-1}$, $-x$ is represented by the binary representation of $2^n - x$

- To compute this: Flip the bits of x then add 1:
  - All 1's string is $2^n - 1$, so

    Flip the bits of x $\equiv$ replace x by $2^n - 1 - x$