PROGRAMOZÁS
tantárgy

Gregorics Tibor
egyetemi docens
ELTE Informatikai Kar

Követelmények

	A,C,E szakirány	B szakirány
Előfeltétel	Prog. alapismeret	Prog. alapismeret Diszkrét matematika I.
Óraszám	2 ea + 2 táblás gy. + 2 labor gy. + 1 konz	2 ea + 2 táblás gy. + 1 konz

Számonkérés	A,C,E szakirány	B szakirány
Plusz-mínusz röpdolgozat	minden héten összege nem lehet negatív	minden héten összege nem lehet negatív
Zárthelyi	**3 db** + 1 javító 3. zh évfolyam szintű két utolsó zh *legalább elégséges*	**3 db** (*duplán számít*) + 1 javító 3. zh évfolyam szintű két utolsó zh *legalább elégséges*
Házi feladat	**2 db**, határidőre, de még a szorgalmi időszakban	**3 db**, határidőre, de még a szorgalmi időszakban,
Géptermi zárthelyi	**2 db** (+ 1 javító) 1. zh 5 fokozatú 2. évfolyam zh 3 fokozatú, *legalább megfelelt*	**1 db** (+ 1 javító) évfolyam zh három fokozatú, *legalább megfelelt*

Segédanyagok

Gregorics Tibor: Programozás – 1.kötet – Tervezés.
ELTE Eötvös Kiadó, 2013.

Gregorics Tibor: Programozás – 2.kötet – Megvalósítás.
ELTE Eötvös Kiadó, 2013.

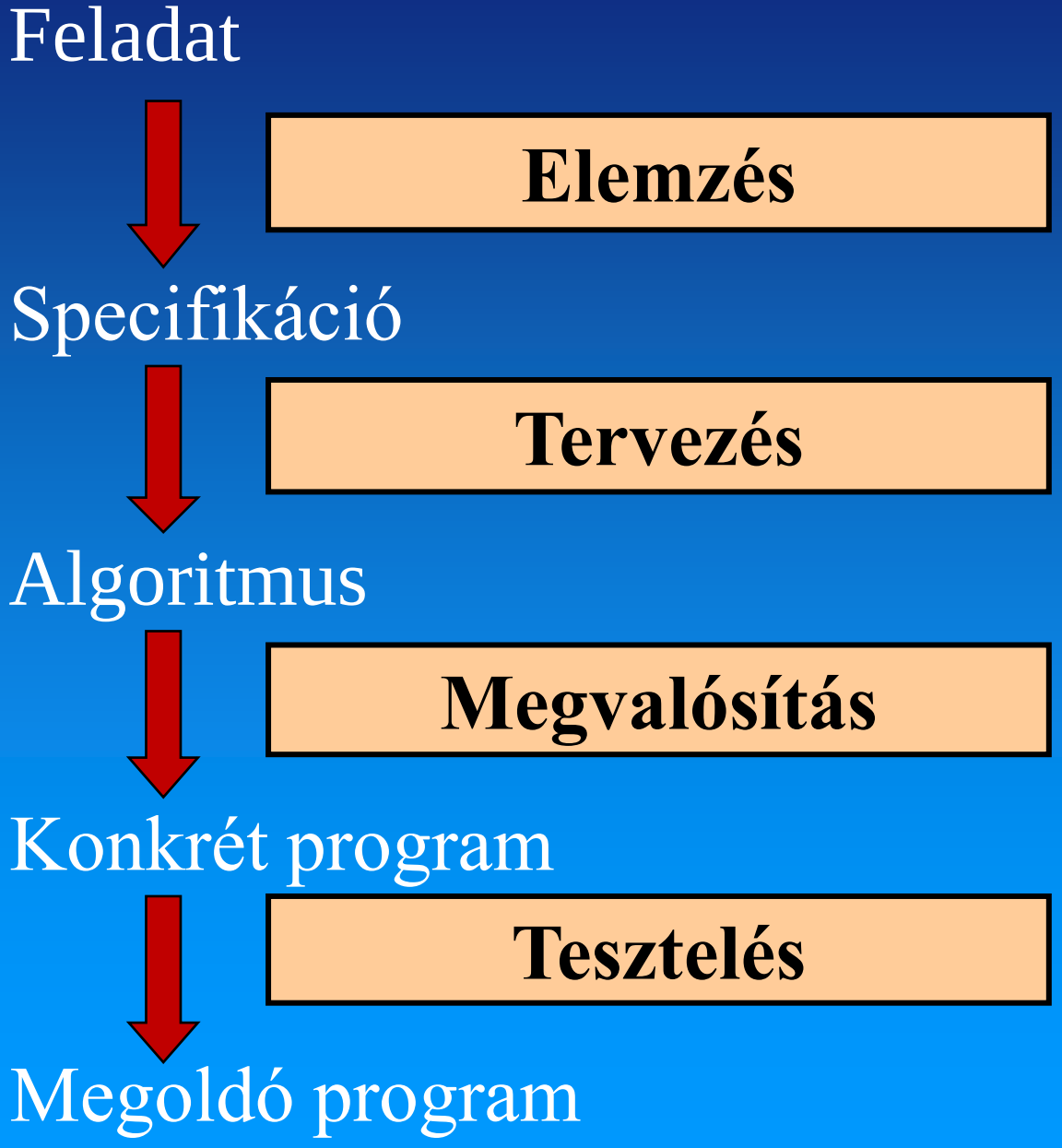
http://people.inf.elte.hu/gt/prog

PROGRAMOZÁS

Programozási szabványok

Gregorics Tibor
http://people.inf.elte.hu/gt/prog

Egyszerű programozási feladat megoldásának fázisai



Néhány programozási szabvány

Specifikálás

- adat központú elő-, utófeltételes leírás ✓
- állapot központú elő-, utófeltételes leírás
- „végrehajtható” leírás

Tervezés

- algoritmikus gondolkodás ✓
- analóg módon (programozási tétellel)
  - analóg algoritmikus gondolkodás ✓
  - visszavezetés
- formális program-szintézis (levezetés)

Megvalósítás

- hármas tagozódás ✓
- top-down kódolás ✓
- beolvasás, kiírás kódolása ✓
- adatellenőrzés beépítése ✓
- adattípusok kiválasztása ✓
- modulokra bontás
- objektum orientált elemek :
  - class
  - öröklés
  - dinamikus kötés
- …

Tesztelés

- fekete/fehér doboz ✓
- érvényes/ érvénytelen ✓
- határ esetek, adat-csoportok szerint ✓
- megvalósított művelet tulajdonságai
- terheléses teszt
- programozási tételekre jellemző teszt

Dokumentálás

- Felhasználói és fejlesztői ✓
- …

Egy szabványokat sértő megoldás

Számoljuk ki két természetes szám szorzatát szorzás nélkül!

Adatokra felírt specifikáció:

Bemenet: $a, b \in \mathbb{Z}$

Eredmény: $c \in \mathbb{Z}$

Előfeltétel: $a \geq 0 \wedge b \geq 0$

Utófeltétel: $c = a \cdot b$

Algoritmus:

Változók:

$a \sim x : \mathbb{Z}$

$b \sim y : \mathbb{Z}$

$c \sim z : \mathbb{Z}$

$z := 0$

$x \neq 0$

$z := z + y$

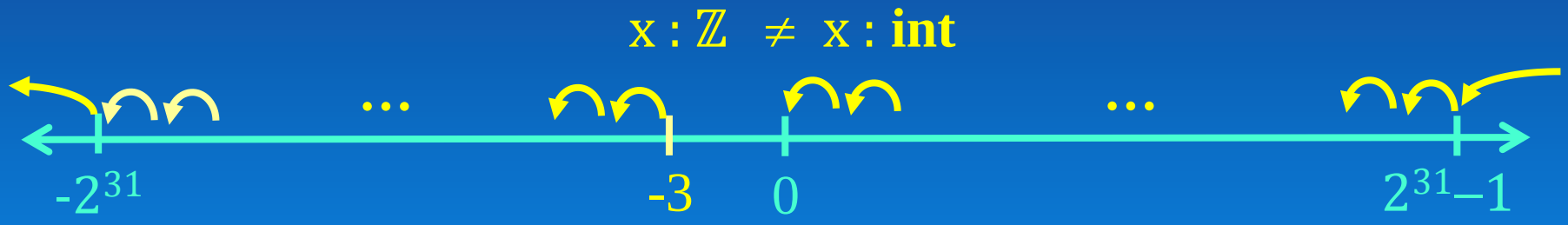
$x := x - 1$

```
int x, y, z;
cout << "x = "; cin >> x;
if(x<0){
   cout << "hiba\n";
   return 1;
}
cout << "y = "; cin >> y;
if(y<0){
   cout << "hiba\n";
   return 1;
}
z = 0;
while(x!=0){
   z = z + y;
   x = x - 1;
}
cout << "x*y = " << z << endl;
```

Amikor egy rosszul megírt program jól működik

Túlcsordulás:

$[-2^{31}-1]_{32} = [-2147483649]_{32} = [2147483647]_{32} = 2^{31}-1$



Ha kezdetben $x=a<0$, akkor a sorozatos $x:=x-1$ hatására $2^{32}-|a|$ lépés múlva lesz $x=0$.

Mindeközben a $z := z + y$ ($y=b$) is $2^{32}-|a|$-szor hajtódik végre, így a z értéke többször is túlcsordulhat.

```
z = 0;
while(x!=0){
   z = z + y;
   x = x - 1;
}
```

Az algoritmus $a<0$ esetén a $(2^{32}-|a|) \cdot b$ értéket számolja ki, és tárolja el a z változóban 32 biten:

$$z = [(2^{32}-|a|) \cdot b]_{32} = [\, a \cdot b + b \cdot 2^{32} \,]_{32} = [a \cdot b]_{32}$$

Specifikáció másképp

$x = x_0 \wedge y = y_0$

$x = x' \wedge y = y'$

x , y , z
egész típusú változók

Kezdetben
x és y input változók
kezdő értéke nem negatív

Állapottér (A) : $x : \mathbb{Z},\ y : \mathbb{Z},\ z : \mathbb{Z}$

Előfeltétel (Ef) : $x = a \wedge y = b \wedge x \geq 0 \wedge y \geq 0$

Utófeltétel (Uf): $z = a \cdot b$

Végezetül a z output változó
az x és y kezdő értékeinek szorzatát
tartalmazza

A: $x : \mathbb{Z},\ y : \mathbb{Z},\ z : \mathbb{Z}$

Ef: $x = a \wedge y = b \wedge x \geq 0 \wedge y \geq 0$

Uf: $y = b \wedge z = a \cdot b$

$z = a \cdot y$

A: $x : \mathbb{Z},\ y : \mathbb{Z}$

Ef: $x = a \wedge y = b \wedge x \geq 0 \wedge y \geq 0$

Uf: $x = a \wedge y = a \cdot b$

input és output
változó egyben

Végrehajtható specifikáció:

A: $x : \mathbb{Z},\ y : \mathbb{Z},\ z : \mathbb{Z}$

Ef: $x = a \wedge y = b \wedge x \geq 0 \wedge y \geq 0$

Uf: $x = a \wedge y = b \wedge z = \sum_{i=1}^{x} y$

$y + y + \ldots + y$

x-szer

Összegzés programozási tétele

Összegezzük az $f : [m..n] \to H$ függvénynek
az m..n egész-intervallumon felvett értékeit!

A H halmazon legyen értelmezett a $+ : H \times H \to H$ művelet,
amely asszociatív és van baloldali nulla eleme ($0 \in H$).

A : $m:\mathbb{Z}, n:\mathbb{Z}, s:H$

Ef : $m=m_0 \wedge n=n_0$

Uf : $Ef \wedge s=\sum_{i=m}^{n} f(i)$

$\sum_{i=m}^{n} f(i) = 0$ ha $n<m$

```
s, i := 0, m
i≦n
    s := s + f(i)
    i := i + 1
```

i : ℤ

```
s := 0
i =m .. n
    s := s+f(i)
```

Visszavezetés módszere

A: $m : \mathbb{Z},\ n : \mathbb{Z},\ s : H$

Ef: $m = m_0 \wedge n = n_0$

Uf: $Ef \wedge s = \sum_{i=m}^{n} f(i)$

A: $x : \mathbb{Z},\ y : \mathbb{Z},\ z : \mathbb{Z}$

Ef: $x = a \wedge y = b \wedge x \geq 0 \wedge y \geq 0$

Uf: $x = a \wedge y = b \wedge z = \sum_{i=1}^{x} y$

$s : H$		$z : \mathbb{Z}$
$s = \sum_{i=m}^{n} f(i)$		$z = \sum_{i=1}^{x} y$
$m \,..\, n$	$\sim$	$1 \,..\, x$
s	$\sim$	z
$f(i)$	$\sim$	y
$H, +, 0$	$\sim$	$\mathbb{Z}, +, 0$

De hiszen ez egy ***ÖSSZEGZÉS***!!!

$z := 0$

$i = 1 \,..\, x$

$z := z + y$

Megvalósítás

Szabványok, amire figyelni kell:

- hármas tagozódás, struktogram top-down kódolása
- adatellenőrzés a specifikáció előfeltétele alapján
- konkrét adattípusok kiválasztása C++-ban: natural →int
  (bool, char, int, double, string, vector<>)
- kódolási konvenciók C++-ban
  - programozási tételek (számlálós ciklus for utasítással, ++i,)
  - csak korlátozottan használjuk a do-while ciklust
  - globális változót ne használjunk
  - kerüljük a kódismétlődést (pl. alprogramok használatával)
- nem funkcionális követelmények:
  - barátságos
  - öndokumentáló (kiírás, komment)
  - bolond-biztos

Tesztelés

- **Specifikáció alapján** (fekete doboz)
  - Érvényes tesztesetek
    - Alkalmazott programozási tétel alapján
      - intervallum: ~hossza (0, 1, sok) ,~ eleje, ~vége
      - tétel specifikus: összegzés esetén a skálázhatóság
    - Értékhalmazok adat-csoportjai és határesetei
    - Speciális műveleti tulajdonságok
  - Érvénytelen tesztesetek
- **Program (algoritmus+kód) alapján** (fehér doboz)
  - Beolvasás tesztelése (bolond-biztos, öndokumentáló)
  - Kiírás tesztelése (öndokumentáló)
  - Minden utasítás, és minden egymás után végrehajtható utasítás-pár kipróbálása. (egyszerű programok esetén a korábbi esetek lefedik)

A példa fekete doboz tesztesetei 1.

- **Összegzés tételéből** adódó tesztesetek:
  - [1..x] **elejének, végének vizsgálata** (egyben) :

feltéve, hogy kezdetben z=0

$x = 0 \longrightarrow z = 0$

$x = 1,\ y = 5 \longrightarrow z = 5$

$x = 2,\ y = 7 \longrightarrow z = 14$

hibaüzenet pontosítása

  - [1..x] **hosszának tesztje**: lásd előbbi tesztadatok
  - Eredmény **skálázása**:

Az `int`-tel ábrázolható legnagyobb egész szám: $2^{31}-1 = 2147483647$

input egyike: $2^{31} \longrightarrow$ hibás input

input: 2, $2^{31}-1 \longrightarrow$ z túlcsordul

input: 1, $2^{31}-1 \longrightarrow z = 2^{31}-1$

…

kell ilyen hibaüzenet is

input: 2, $2^{31}-2 \longrightarrow$ z túlcsordul

input: 1, $2^{31}-2 \longrightarrow z = 2^{31}-2$

…

végtelen ciklus javítása

```
for(int i=1; i<=x; ++i)
```

helyett

```
for(int i=0; i<x; ++i)
```

input: 2, $2^{30} \longrightarrow$ z túlcsordul

input: 1, $2^{30} \longrightarrow z = 2^{30}$

A példa fekete doboz tesztesetei 2.

- **Adattípusok értékeinek jellegzetes csoportjai** (egész számoknál a nulla, negatívok és pozitívok), határ-elemei (természetes számoknál a nulla)
  - mindkettő input (x és y) nulla: $0 \cdot 0$ $\longrightarrow z = 0$
  - csak az egyik input (x vagy y) nulla: $12 \cdot 0, 0 \cdot 12$ $\longrightarrow z = 0$
  - egyik input sem nulla: $12 \cdot 7$ $\longrightarrow z = 84$
- A megvalósított **műveletek tulajdonságai** (szorzás):
  - kommutatív: $12 \cdot 7 = 7 \cdot 12$ $\longrightarrow z = 84$
  - asszociatív: $(12 \cdot 7) \cdot 5 = 12 \cdot (7 \cdot 5)$ $\longrightarrow z = 420$
  - null elem: $12 \cdot 0 = 0 \cdot 12$ $\longrightarrow z = 0$
  - egység elem: $12 \cdot 1 = 1 \cdot 12$ $\longrightarrow z = 12$

futási idő eltér

- **Érvénytelen input**:
  - előfeltétel sérül: $x<0$ vagy $y<0$ $\longrightarrow$ hibás input
    ($x<0, y\geqq 0$ / $x \geqq 0, y<0$ / $x<0, y<0$)
  - számábrázolási korlátok: $x \geqq 2^{31}$ vagy $y \geqq 2^{31}$ $\longrightarrow$ hibás input
    ($x<2^{31}, y\geqq 2^{31}$ / $x\geqq 2^{31}, y<2^{31}$ / $x<2^{31}, y<2^{31}$)

A félév során használt minták, ökölszabályok (szabványok)

- Specifikációs minták
  - A specifikáció formája , és a végrehajtható utófeltétel.
- Tervezési minták
  - Algoritmus tervezéshez programozási tételek, és azok felhasználása visszavezetéssel
  - Objektum-orientált tervezési minták
- Implementációs minták
  - Implementációs stratégiák (hármas tagozódás, kívülről befele)
  - Kódminták (beolvasás, programozási tételek kódjai), kódolási ajánlások (konvenciók) alkalmazása
- Tesztelési stratégiák
  - Általános szempontok (fekete-fehér, érvényes-érvénytelen)
  - Programozási tételek tesztesetei (intervallum, tétel sajátossága)
  - Adat-csoportok (határ esetek, műveletek tulajdonságai)
- Dokumentálási minták