|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Neptun kód: | **ABC123** |  | Név: | **Készítő Neve** |
| Beadás verziószáma: | 1.  Később kiegészítéssel(!)  2. (javított Uf…)  3. (javított Algoritmus) | | | |

## Feladat

A feladatot – a végleges komplex beadandó kivételével – képként is be lehet tenni. Jelenjen meg a feladat teljes szövege, a minta bemenettel, kimenettel együtt. (A komplex beadandó végleges verziójában figyeljünk arra, hogy a „Készítsen programot, amely…” jellegű mondat értelme az, hogy a programunknak az a feladata, hogy programot készítsen. Helyesen: a program az „amely…”-ben megadott feladatot oldja meg.)

A képként betett feladat tartalmazza a Bemenet és Kimenet leírását, ezt a komplex beadandóban külön le kell írni a megfelelő részeknél, részletezve a kézi és biro-s beolvasást és kiírást.

## Adatreprezentáció

A kiadott feladat mintaadata segíti a feladat megértését, ez egy esete, reprezentációja a bemenet-kimenet kapcsolatának. Már az 1. fázisban adjunk meg több példát/esetet/adatreprezentációt! Ezeket a specifikációban is használjuk tesztként és a kész programot is teszteljük ugyanezekkel az adatokkal.

A specifikációnál megadott adatreprezentációknál is érdemes gondolni a kis elemszámú, érvényes triviális esetekre, a kezelhető, 5-20 bemenő adatú érvényes, nem triviális esetekre és az előfeltétel teljesülését vizsgáló érvénytelen esetekre. A triviális esetek között legyen olyan is, ami a „ha nincs, akkor … legyen az eredmény” típusú megoldást ellenőrzi. Példa p/q típusú tört értékének kiszámítására:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1. | 2. | 3. | 4. | 5. |
| #bemenet  a: 4  b: 5  #kimenet  x: 0.8 | #bemenet  a: 4  b: 3  #kimenet  x: 1.3 | #bemenet  a: 4  b: 3  #kimenet  x: 1.3333333333 | #bemenet  a: -4  b: 1  #kimenet  x: -4 | #bemenet  a: -4  b: 0  #kimenet  x: 4 |
| helyes | hibás | helyes | helyes | előfeltétel hiba |

A példában a 2. eset a túl nagy kerekítés miatt hibás, míg a 3. „kellően jó közelítés”, ami adódhat a tesztkörnyezetből is vagy abból, hogy az utófeltétel kitér a pontosságra, pl. abs(x\*b-a)<0.005. Az 5. eset („precondition is not true”) az előre megadott feltétel b<>0 védi a műveletek elvégzésétől.

## Specifikáció

Megelőzheti az adatreprezentációt. Egyszerű formátummal a progalap.inf.elte.hu/specifikacio oldalról átmásolt kód, de illik „szépíteni”. A szóköz többszörözést tabulátorokra cserélhetjük. a „->” lecserélhető a → jelre. A szép megjelenést a Cambria Math betűtípus biztosítja, a szép tördeléshez a tabulátorokat 0,5–1 cm sűrűségűre érdemes állítani, ez az egész dokumentumra érvényes lesz. Hosszabb részeket sortöréssel érdemes több sorba rendezni, így függő behúzással és bekezdés utáni térközzel biztosítható a kód átláthatósága. Például:

Be: a ∈ Z, b ∈ Z

Ki: x ∈ R

Ef: b <> 0

Uf: b = 1 → x = a és  
b = -1 → x = -a és   
abs(b) <> 1 → abs(b \* x – a) < 0.005

**Link:** A progalap.inf.elte.hu/specifikacio oldalon *Save*-vel (vagy *Share*-rel)vágólapra mentett hivatkozás. (Figyelve arra, hogy módosítást követően a kódot és ezt a linket is frissíteni kell.

## Programozási minta

A használt programozási mintát csak az első fázisban kell megjeleníteni. A minta az általános képlet, amibe behelyettesítjük a konkrét feladat megoldásának paramétereit. (Hasonlóan a lineáris függvények y=mx+b alakja „minta”, általános képlet, az y=2x+3 egy konkrét példa)

Egy feladattípus megoldásához sokféle módszer lehetséges. Itt azonban mindig a tananyagban szereplő mintát kell használni, meg kell adni a programozási minta elérésének adatait. (Pontosan.)

**Forrás:   
Részlet:**

## Visszavezetés

Az alkalmazott programozási minta alapján a specifikáció és az algoritmus elemei közötti kapcsolatot adja meg. Célszerű – olvashatóbb –, ha minden elemet felveszünk a visszavezetésbe, mert egyes elemek elhagyása félreértést okozhat. Például, ha elhagyjuk az i ~ 1-t és a minért ~  – elemeket, akkor nem tudjuk, hogy a minért-re nincs szükségünk, vagy az i-hez hasonlóan az alkalmazás során a sablonban szereplővel azonos nevű adatról van szó.

A visszavezetés megjelenésének stílusa lehet azonos a specifikáció stílusával. Főleg több tétel visszavezetéséhez célszerű az egyes táblákat szövegdobozban táblázatban elhelyezni. Például:

…

Maximum-kiválasztás

|  |  |  |
| --- | --- | --- |
| i | ~ | i |
| e..u | ~ | 1..n |
| f(i) | ~ | a[i] |
| maxind | ~ | maxi |
| maxért | ~ | - |

Minimum-kiválasztás

|  |  |  |
| --- | --- | --- |
| i | ~ | i |
| e..u | ~ | 1..n |
| f(i) | ~ | a[i] |
| minind | ~ | mini |
| minért | ~ | - |

## Algoritmus

Szerkesztés a gyakorlatokon is használt algoritmus-rajzoló alkalmazás progalap.inf.elte.hu/stuki alkalmazásával történjen. A kép képernyőkivágással, a „Copy png” vagy a „Save png” menüpontokkal menthető és illeszthető be a dokumentumba.

Bármilyen módon készül az algoritmus, az eredményt nyomtatási minőségben, általában nyomtatási képből jelenítjük meg.

Növeli az érthetőséget, ha függvényekre bontva, pontos függvénydeklarációval adunk algoritmust, nem csak a „szignifikáns részleteket” vetjük papírra. A függvény/eljárás pontos megadásával magától értetődővé válik az adatok algoritmusban használandó formátumú deklarációja, a feladatspecifikus típusok, globális és lokális változók, paraméterek megjelenítése.

Az algoritmus struktúrájának helyes megjelenítése mellett az is számít, hogy az ábrán minden olvasható elég nagy és tiszta kontúrú legyen, ugyanakkor kompakt méretben jelenjen meg.

**Link:** A progalap.inf.elte.hu/stuki oldalon *Save*-vel (vagy *Share*-rel)vágólapra mentett hivatkozás. (Figyelve arra, hogy módosítást követően a kódot és ezt a linket is frissíteni kell.