Informatika 3.
1. előadás: Bevezető

Kovács Kristóf

Budapesti Műszaki és Gazdaságtudományi Egyetem

2024-02-13

Követelmények

- 2 ZH
  - április 12., május 24.
  - egyenként 50 pontot érnek
  - egyenként 40% teljesítése követelmény (ZH-nként)
  - ponthatárok: 40,55,70,85

Követelmények

- 2 ZH
  - április 12., május 24.
  - egyenként 50 pontot érnek
  - egyenként 40% teljesítése követelmény (ZH-nként)
  - ponthatárok: 40,55,70,85
- heti HF
  - 2-2 pontot érnek
  - 40% teljesítése követelmény (8 pont)
  - követelmény feletti pontokkal lehet javítani a végleges jegyen (max 12 pontot)

Követelmények

- 2 ZH
  - április 12., május 24.
  - egyenként 50 pontot érnek
  - egyenként 40% teljesítése követelmény (ZH-nként)
  - ponthatárok: 40,55,70,85
- heti HF
  - 2-2 pontot érnek
  - 40% teljesítése követelmény (8 pont)
  - követelmény feletti pontokkal lehet javítani a végleges jegyen (max 12 pontot)
- gyakorlaton jelenlét/kisZH
  - gyakorlatok elején, az előadás anyagából, 1-2 perces kérdések
  - 1-1 pontot érnek
  - 40% teljesítése követelmény (4 pont)
  - követelmény feletti pontokkal lehet javítani a végleges jegyen (max 6 pontot)

Követelmények

- 2 ZH
  - április 12., május 24.
  - egyenként 50 pontot érnek
  - egyenként 40% teljesítése követelmény (ZH-nként)
  - ponthatárok: 40,55,70,85
- heti HF
  - 2-2 pontot érnek
  - 40% teljesítése követelmény (8 pont)
  - követelmény feletti pontokkal lehet javítani a végleges jegyen (max 12 pontot)
- gyakorlaton jelenlét/kisZH
  - gyakorlatok elején, az előadás anyagából, 1-2 perces kérdések
  - 1-1 pontot érnek
  - 40% teljesítése követelmény (4 pont)
  - követelmény feletti pontokkal lehet javítani a végleges jegyen (max 6 pontot)
- $Z_1 + Z_2 + \max(0, \min(H, 20) - 8) + \max(0, \min(K, 10) - 4) + ??$

Feltérképezés

Példák

1. Adjunk algoritmust mely egy bemeneti listából kiválogatja a páros számokat és visszaadja ezek listáját.

Feltérképezés

Példák

1. Adjunk algoritmust mely egy bemeneti listából kiválogatja a páros számokat és visszaadja ezek listáját.
2. Találjunk ki adatszerkezetet ami komplex számokat tárol.

Feltérképezés

Példák

1. Adjunk algoritmust mely egy bemeneti listából kiválogatja a páros számokat és visszaadja ezek listáját.
2. Találjunk ki adatszerkezetet ami komplex számokat tárol.

Feladatok

1. Adjunk algoritmust mely visszaadja a 100-nál kisebb 7-el osztható számokat.

Feltérképezés

Példák

1. Adjunk algoritmust mely egy bemeneti listából kiválogatja a páros számokat és visszaadja ezek listáját.
2. Találjunk ki adatszerkezetet ami komplex számokat tárol.

Feladatok

1. Adjunk algoritmust mely visszaadja a 100-nál kisebb 7-el osztható számokat.
2. Adjunk algoritmust mely visszaadja a 100-nál kisebb prím számokat.

Feltérképezés

Példák

1. Adjunk algoritmust mely egy bemeneti listából kiválogatja a páros számokat és visszaadja ezek listáját.
2. Találjunk ki adatszerkezetet ami komplex számokat tárol.

Feladatok

1. Adjunk algoritmust mely visszaadja a 100-nál kisebb 7-el osztható számokat.
2. Adjunk algoritmust mely visszaadja a 100-nál kisebb prím számokat.
3. Adjunk algoritmust mely visszaadja az első 100 prím számot.

Feltérképezés

Példák

1. Adjunk algoritmust mely egy bemeneti listából kiválogatja a páros számokat és visszaadja ezek listáját.
2. Találjunk ki adatszerkezetet ami komplex számokat tárol.

Feladatok

1. Adjunk algoritmust mely visszaadja a 100-nál kisebb 7-el osztható számokat.
2. Adjunk algoritmust mely visszaadja a 100-nál kisebb prím számokat.
3. Adjunk algoritmust mely visszaadja az első 100 prím számot.
4. Adjunk algoritmust mely egy bemeneti listából kiválogatja a 7-el osztható számokat.

Feltérképezés

Példák

1. Adjunk algoritmust mely egy bemeneti listából kiválogatja a páros számokat és visszaadja ezek listáját.
2. Találjunk ki adatszerkezetet ami komplex számokat tárol.

Feladatok

1. Adjunk algoritmust mely visszaadja a 100-nál kisebb 7-el osztható számokat.
2. Adjunk algoritmust mely visszaadja a 100-nál kisebb prím számokat.
3. Adjunk algoritmust mely visszaadja az első 100 prím számot.
4. Adjunk algoritmust mely egy bemeneti listából kiválogatja a 7-el osztható számokat.
5. Adjunk algoritmust mely egy bemeneti listából kiválogatja a prím számokat.

Feltérképezés

Adjunk algoritmust mely...

1. tetszőleges hosszú folyamatosan érkező számoknak minden szám után visszaadja az eddigiek átlagát.

Feltérképezés

Adjunk algoritmust mely...

1. tetszőleges hosszú folyamatosan érkező számoknak minden szám után visszaadja az eddigiek átlagát.
2. (n,m) párokat tartalmazó bemeneti listából kiválogatja azokat melyek relatív prím párok.

Feltérképezés

Adjunk algoritmust mely...

1. tetszőleges hosszú folyamatosan érkező számoknak minden szám után visszaadja az eddigiek átlagát.
2. (n,m) párokat tartalmazó bemeneti listából kiválogatja azokat melyek relatív prím párok.
3. szavakat tartalmazó bemeneti listában megszámolja hány szóban szerepel legalább egyszer a c karakter.

Feltérképezés

Adjunk algoritmust mely...

1. tetszőleges hosszú folyamatosan érkező számoknak minden szám után visszaadja az eddigiek átlagát.
2. (n,m) párokat tartalmazó bemeneti listából kiválogatja azokat melyek relatív prím párok.
3. szavakat tartalmazó bemeneti listában megszámolja hány szóban szerepel legalább egyszer a c karakter.
4. szavakat tartalmazó bemeneti listában megkeresi hol szerepel a c karakter és visszaadja (szó, index) párok listájaként.

Feltérképezés

Adjunk algoritmust mely...

1. tetszőleges hosszú folyamatosan érkező számoknak minden szám után visszaadja az eddigiek átlagát.
2. (n,m) párokat tartalmazó bemeneti listából kiválogatja azokat melyek relatív prím párok.
3. szavakat tartalmazó bemeneti listában megszámolja hány szóban szerepel legalább egyszer a *c* karakter.
4. szavakat tartalmazó bemeneti listában megkeresi hol szerepel a *c* karakter és visszaadja (szó, index) párok listájaként.
5. bemeneti két komplex számnak (real, imag adattagok) visszaadja a szorzatát.

Feltérképezés

Adjunk algoritmust mely...

1. tetszőleges hosszú folyamatosan érkező számoknak minden szám után visszaadja az eddigiek átlagát.
2. (n,m) párokat tartalmazó bemeneti listából kiválogatja azokat melyek relatív prím párok.
3. szavakat tartalmazó bemeneti listában megszámolja hány szóban szerepel legalább egyszer a c karakter.
4. szavakat tartalmazó bemeneti listában megkeresi hol szerepel a c karakter és visszaadja (szó, index) párok listájaként.
5. bemeneti két komplex számnak (real, imag adattagok) visszaadja a szorzatát.
6. bemeneti két objektumon elvégzi a $*$ műveletet és visszaadja az eredményét.

Feltérképezés

Adjunk algoritmust mely...

1. tetszőleges hosszú folyamatosan érkező számoknak minden szám után visszaadja az eddigiek átlagát.
2. (n,m) párokat tartalmazó bemeneti listából kiválogatja azokat melyek relatív prím párok.
3. szavakat tartalmazó bemeneti listában megszámolja hány szóban szerepel legalább egyszer a *c* karakter.
4. szavakat tartalmazó bemeneti listában megkeresi hol szerepel a *c* karakter és visszaadja (szó, index) párok listájaként.
5. bemeneti két komplex számnak (real, imag adattagok) visszaadja a szorzatát.
6. bemeneti két objektumon elvégzi a $*$ műveletet és visszaadja az eredményét.
7. bemeneti objektumon párokat tartalmazó listában elvégzi páronként a / műveletet és visszaadja az eredményét, figyelve arra, hogy ha hiba történik (nem sikerül a /) akkor az *err* szót helyettesítse az eredmény helyére.

Feltérképezés

Adjunk adatszerkezet(ek)et mely(ek)...

1. képes reprezentálni egy négyzetet 2 dimenziós térben.

Feltérképezés

Adjunk adatszerkezet(ek)et mely(ek)...

1. képes reprezentálni egy négyzetet 2 dimenziós térben.
2. képes reprezentálni egy tetszőleges sokszöget 2 dimenziós térben.

Feltérképezés

Adjunk adatszerkezet(ek)et mely(ek)...

1. képes reprezentálni egy négyzetet 2 dimenziós térben.
2. képes reprezentálni egy tetszőleges sokszöget 2 dimenziós térben.
3. egy egyszerű lista, de tárolja hányszor helyeztünk bele és hányszor kérdeztünk le elemet.

Feltérképezés

Adjunk adatszerkezet(ek)et mely(ek)...

1. képes reprezentálni egy négyzetet 2 dimenziós térben.
2. képes reprezentálni egy tetszőleges sokszöget 2 dimenziós térben.
3. egy egyszerű lista, de tárolja hányszor helyeztünk bele és hányszor kérdeztünk le elemet.
4. képesek reprezentálni kört, háromszöget, téglalapot.

Feltérképezés

Adjunk adatszerkezet(ek)et mely(ek)...

1. képes reprezentálni egy négyzetet 2 dimenziós térben.
2. képes reprezentálni egy tetszőleges sokszöget 2 dimenziós térben.
3. egy egyszerű lista, de tárolja hányszor helyeztünk bele és hányszor kérdeztünk le elemet.
4. képesek reprezentálni kört, háromszöget, téglalapot.
5. képesek reprezentálni kört, háromszöget, téglalapot és egységes terület és kerület számítás függvényük létezik, melyet meg lehet hívni anélkül hogy tudnánk melyik alakzatot tárolja ezek közül az objektumunk.

Történet

C

- Dennis Ritchie fejlesztett ki
- régi (1972), de a mai napig nagyon sokat használt
- fordított nyelv (nem interpretált, mint a python)
- hatékony, nagyon közel van a gépi kódhoz
- könnyedén olvashatatlanná tehető (nem feltétlen szándékosan)

Történet

C

- Dennis Ritchie fejlesztett ki
- régi (1972), de a mai napig nagyon sokat használt
- fordított nyelv (nem interpretált, mint a python)
- hatékony, nagyon közel van a gépi kódhoz
- könnyedén olvashatatlanná tehető (nem feltétlen szándékosan)

C++

- Bjarne Stroustrup fejlesztette
- régi (1985), nagyon elterjedten használt
- tekinthető a C kiegészítésének
- a mai napig fejlesztett, legutóbbi szabvány C++20 (2020)

C-vel kezdünk

```
#include <stdio.h>
int main(void) {
    printf("Hello, World!\n");
    return 0;
}
```

Mielőtt le tudnánk futtatni, le kell fordítanunk.
Fordítók:

- Windows: mingw, visual studio
- Linux: gcc/g++
- Mac: Xcode -> gcc/g++, cc

C-vel kezdünk

```
#include <stdio.h>
int main(void) {
    printf("Hello, World!\n");
    return 0;
}
```

Mielőtt le tudnánk futtatni, le kell fordítanunk.
Fordítók:

- Windows: mingw, visual studio
- Linux: gcc/g++
- Mac: Xcode -> gcc/g++, cc
- stackoverflow-n keresés: *OS C compiler*

Pythonról C-re

- meg kell adnunk a változók típusát, a függvények visszatérési értékének a típusát:

```
int i = 0;
float f = 0.0;
```

Pythonról C-re

- meg kell adnunk a változók típusát, a függvények visszatérési értékének a típusát:

```
int i = 0;
float f = 0.0;
```

- a program kiindulási pontja a *main* függvény:

```
int main(void) {
```

Pythonról C-re

- meg kell adnunk a változók típusát, a függvények visszatérési értékének a típusát:

```
int i = 0;
float f = 0.0;
```

- a program kiindulási pontja a *main* függvény:

```
int main(void) {
```

- a *for* ciklus nem listán iterál:

```
int i;
for(i = 0; i < 10; i++) {
```

Pythonról C-re

- meg kell adnunk a változók típusát, a függvények visszatérési értékének a típusát:

```
int i = 0;
float f = 0.0;
```

- a program kiindulási pontja a *main* függvény:

```
int main(void) {
```

- a *for* ciklus nem listán iterál:

```
int i;
for(i = 0; i < 10; i++) {
```

- nincsenek kényelmes beépített adatszerkezetek (lista, szótár):

```
int t[10];
t[0] = 1;
```