

2. Egy kör esetében tároljuk a kör középpontjának valós koordinátáit (abszcissza és ordináta), a xOy derékszögű koordinátarendszerben. A $c1x$ és $c1y$ változók egy kör középpontjának abszcisszáját és ordinátáját, míg $c2x$ és $c2y$ változók egy másik kör középpontjának abszcisszáját és ordinátáját tárolják. Deklarálja megfelelően a négy változót, és írjon egy utasítássorozatot, melynek végrehajtása után a képernyőn megjelennek, szóközzel elválasztva azon kör középpontjának koordinátái, melynek a középpontja az abszcissza tengelyen helyezkedik el. Ha mindkét kör középpontja az abszcissza tengelyen helyezkedik el, akkor az **ambe1e** kifejezés, míg ha egyik kör középpontja sem az abszcissza tengelyen helyezkedik el, akkor a **niciunul** kifejezés jelenjen meg a képernyőn. **(6p.)**
3. Az **a** változó **char** típusú, míg az **i** és **k** változók egész típusúak; a billentyűzetről beolvasunk 16 nagybetűt az angol abc nagybetűi közül. Anélkül, hogy más változót használnának, írjanak urasításokat a kipontozott részek helyére úgy, hogy az utasítássor végrehajtása után a **k** változó a **{A, C, E, I}** halmaz elemein kívüli, beolvasott nagybetűkarakterek számát tartalmazza!

Példa: ha az alábbi karakterek olvassuk be a billentyűzetről, akkor a **k** változóban a 8-as érték fog megjelenni.

P, R, A, S, L, E, A, C, E, L, V, O, I, N, I, C
(6p.)

```
k=.....  
for(i=1;i<=16;i++)  
{ cin>>a; | scanf("%c",&a);  
  .....  
}
```

III. TÉTEL

(30 pont)

1. Beolvasunk két természetes számot, **n** és **c** ($c \leq 9$), és egy **n**-ből szerkesztett számot íratunk ki úgy, hogy eltávolítjuk belőle az össze **c**-vel megegyező számjegyet vagy **-1**-et íratunk ki, ha **n** összes számjegye megegyezik **c**-vel. A jelentéktelen nulla számjegyeket figyelmen kívül hagyjuk, ahogy a példában is látható.
Írja meg pszeudokódban a feladat megoldási algoritmusát.
Példa: ha **n=50752** vagy **n=72** és **c=5**, akkor **72** lesz kiírva, ha **n=500** és **c=5**, akkor **0** lesz kiírva, viszont ha **n=55** és **c=5**, akkor **-1** íródik ki. **(10p.)**
2. Írjon C/C++ programot, amely billentyűzetről beolvas két természetes számot, **n** és **k** ($n \in [3, 20]$, $k \in [2, n]$), és felépít a memóriában egy **n** elemből álló egydimenziós tömböt a következő tulajdonságokkal:
- a **k**-ik elem értéke 0;
 - a tömb az első pozíciótól a **k**-ik elemig, azt is beleértve, balról jobbra egy szigorúan csökkenő egymás utáni számokból álló számsorozatot tartalmaz, míg a **k**-ik elemtől az utolsó pozícióig, balról jobbra egy szigorúan növekvő egymás utáni számokból álló számsorozatot tartalmaz.
- A program írja ki a képernyőre a felépített tömb elemeit egy-egy szóközzel elválasztva.
Példa: ha **n=10** és **k=3**, akkor a kiírt tömb: (2, 1, 0, 1, 2, 3, 4, 5, 6, 7). **(10p.)**
3. A **bac.in** szöveges állomány egy legalább három és legfeljebb 10^5 nem nulla egész számból álló sorozatot tartalmaz a $[-10^9, 10^9]$ intervallumból, amelyek közül kettő negatív és a többi pozitív. A számok egy-egy szóközzel vannak elválasztva. Egy szekvencia a sorozatban egymás utáni pozíciókon elhelyezkedő tagokból tevődik össze, míg a szekvencia hossza megegyezik a benne szereplő tagok számával.
Írassa ki a képernyőre annak az adott sorozat azon szekvenciájának a hosszát, amely egyetlen negatív és maximális számú pozitív elemet tartalmaz!
Tervezen a futási idő és a memóriahasználat szempontjából hatékony algoritmust!
Példa: ha az állomány a következő számokat tartalmazza: 15 21 -61 9 870 -23 11 5 8, akkor a képernyőn megjelenő érték **6** (a megfelelő szekvencia pedig 9 870 -23 11 5 8).
a. Írja le saját szavaival a használt algoritmust, és indokolja annak hatékonyságát! **(2p.)**
b. Írja meg a tervezett algoritmusnak megfelelő C/C++ programot! **(8p.)**