1. Definirati i inicijalizirati (pomoću inicijalizatora pri definiciji polja, uz nastojanje da se čim manji broj vrijednosti u inicijalizatoru navodi eksplicitno) dvodimenzijsko realno polje veličine 5 redaka i 7 stupaca. Zatim polje ispisati u obliku prikazanom u primjeru izvršavanja programa. Vrijednosti u inicijalizatoru neka budu one koje su prikazane u primjeru.

   Primjer izvršavanja programa

```
··0.0··0.0··0.0··0.0··0.0··0.9··0.0↵
··0.0··0.0··0.0··0.0··0.0··0.0··0.0↵
·31.1·32.2·33.3·34.2··0.0··0.0··0.0↵
··1.0··4.0··0.0··0.0··0.0··0.0··7.7↵
··0.0··0.0··0.0··0.0··0.0··0.0··0.0↵
```

2. Definirati trodimenzijsko cjelobrojno polje veličine 4 sloja, 3 retka i 6 stupaca te ga na najlakši mogući način napuniti vrijednostima prikazanim u primjeru izvršavanja programa. Zatim polje ispisati u obliku prikazanom u primjeru. Uočiti da se iza zadnjeg člana polja koji je ispisan, ispisao samo jedan znak za skok u novi red, a da su slojevi međusobno razdvojeni praznim retkom.

   Primjer izvršavanja programa

```
1. sloj
··111··112··113··114··115··116↵
··121··122··123··124··125··126↵
··131  132··133··134··135··136↵
↵
2. sloj
··211··212··213··214··215··216↵
··221··222··223··224··225··226↵
··231··232··233··234··235··236↵
↵
3. sloj
··311··312··313··314··315··316↵
··321··322··323··324··325··326↵
··331··332··333··334··335··336↵
↵
4. sloj
··411··412··413··414··415··416↵
··421··422··423··424··425··426↵
··431··432··433··434··435··436↵
```

3. Učitati vrijednosti za broj redaka m i broj stupaca n dvodimenzijskog cjelobrojnog polja. Ne treba provjeravati jesu li upisane ispravne vrijednosti. Po retcima učitati vrijednosti članova dvodimenzijskog cjelobrojnog polja od m redaka i n stupaca. Nakon što su sve vrijednosti upisane, pronaći i ispisati **indekse** najmanjih članova u svakom **stupcu**.

Primjer izvršavanja programa

```
Upisite m, n > 4 5↵
Upisite 4 x 5 clanova >↵
1 2 7 4 2↵
4 3 2 1 3↵
1 3 2 1 1↵
4 0 2 0 2↵
1. stupac: (0, 0), (2, 0)↵
2. stupac: (3, 1)↵
3. stupac: (1, 2), (2, 2), (3, 2)↵
4. stupac: (3, 3)↵
5. stupac: (2, 4)↵
```

4. Učitati vrijednosti za broj redaka m i broj stupaca n dvodimenzijskog cjelobrojnog polja. Ne treba provjeravati jesu li upisane ispravne vrijednosti. Po retcima učitati vrijednosti članova dvodimenzijskog cjelobrojnog polja od m redaka i n stupaca. Nakon toga treba retke matrice posmaknuti prema gore: redak s indeksom **i** dobiva vrijednosti iz retka s indeksom **i+1**, a posljednji redak polja dobiva vrijednosti retka s indeksom 0. Nakon obavljenog posmaka redaka, ispisati novu matricu. Zadatak treba riješiti bez upotrebe pomoćnog polja.

Primjer izvršavanja programa

```
Upisite m, n > 5 2↵
Upisite 5 x 2 clanova >↵
1 2↵
4 3↵
1 3↵
4 0↵
9 7↵
Nakon posmaka:↵
···4···3↵
···1···3↵
···4···0↵
···9···7↵
···1···2↵
```

5. Generirati kvadratnu matricu dimenzija 11 x 11 kojoj su svi elementi glavne i sporedne dijagonale, elementi prvog i zadnjeg retka, te prvog i zadnjeg stupca postavljeni na vrijednost 1, a svi ostali elementi matrice postavljeni na vrijednost 8. Generiranu matricu ispisati na zaslon.

Primjer izvršavanja programa

```
Generirana kvadratna matrica:↵
·1·1·1·1·1·1·1·1·1·1·1↵
·1·1·8·8·8·8·8·8·8·1·1↵
·1·8·1·8·8·8·8·8·1·8·1↵
·1·8·8·1·8·8·8·1·8·8·1↵
·1·8·8·8·1·8·1·8·8·8·1↵
·1·8·8·8·8·1·8·8·8·8·1↵
·1·8·8·8·1·8·1·8·8·8·1↵
·1·8·8·1·8·8·8·1·8·8·1↵
·1·8·1·8·8·8·8·8·1·8·1↵
·1·1·8·8·8·8·8·8·8·1·1↵
·1·1·1·1·1·1·1·1·1·1·1↵
```

6. Uz kontrolu učitati vrijednosti za broj redaka m i broj stupaca n dvodimenzijskog cjelobrojnog polja. Broj redaka mora biti iz intervala [4, 8], a broj stupaca iz intervala [5, 10]. Nakon toga učitati članove matrice. Jednodimenzijsko polje sumaPoStupcima napuniti sumama vrijednosti po stupcima. Nakon toga ispisati članove polja sumaPoStupcima.

Primjer izvršavanja programa

```
Upisite m [4, 8] > 3↵
Upisite m [4, 8] > 20↵
Upisite m [4, 8] > 5↵
Upisite n [5, 10] > 6↵
Upisite 5 x 6 clanova >↵
1 2 3 4 5 6↵
1 3 1 2 1 1↵
9 1 2 2 5 0↵
4 7 8 0 0 0↵
9 2 3 3 7 4↵
Sume po stupcima:↵
···24···15···17···11···18···11
```

7. Učitati vrijednosti za broj redaka m i broj stupaca n dvodimenzijskog cjelobrojnog polja. Ne treba provjeravati jesu li upisane ispravne vrijednosti. Po retcima učitati vrijednosti članova dvodimenzijskog cjelobrojnog polja od m redaka i n stupaca. Sortirati vrijednosti unutar svakog pojedinog stupca, od manjih prema većim, a zatim polje ispisati.

Primjer izvršavanja programa

```
Upisite m, n > 5 3↵
Upisite 5 x 3 clanova >↵
9 2 0↵
4 3 -1↵
1 3 -9↵
3 8 6↵
8 1 3↵
Nakon sortiranja stupaca:↵
   1   1  -9↵
   3   2  -1↵
   4   3   0↵
   8   3   3↵
   9   8   6↵
```

8. Pretinci u sefu označeni su šiframa (7-znamenkasti pozitivni brojevi). Za svaki pretinac, osim šifre, evidentiraju se njegove mjere izražene u centimetrima: visina, širina i dubina (cijeli brojevi), te volumen u litrama (realni broj) koji se izračunava na temelju visine, širine i dubine pretinca.

   Učitati cijeli broj n koji predstavlja broj pretinaca koje treba evidentirati, a zatim za svaki od n pretinaca učitati šifru pretinca, njegovu visinu, širinu i dubinu, a volumen izračunati i pohraniti u podatke tog pretinca. Nakon toga, u poretku od najvećih prema najmanjim pretincima, ispisati šifru pretinca i njegov volumen izražen u litrama.

Primjer izvršavanja programa

```
Upisite broj pretinaca > 7↵
Upisite podatke za pretince (7)↵
 1. pretinac > 1234567 12 50 12↵
 2. pretinac > 3122222 10 22 15↵
 3. pretinac > 2001100 20 3 5↵
 4. pretinac > 9122121 5 30 20↵
 5. pretinac > 3312111 102 55 31↵
 6. pretinac > 5210001 20 10 15↵
 7. pretinac > 4000001 19 27 62↵
Sortirani pretinci:↵
3312111 = 173.91 litara↵
4000001 =  31.81 litara↵
1234567 =   7.20 litara↵
3122222 =   3.30 litara↵
5210001 =   3.00 litara↵
9122121 =   3.00 litara↵
2001100 =   0.30 litara↵
```

Rješenja:

1.
```
#include <stdio.h>

#define BR_REDAKA 5
#define BR_STUPACA 7

int main(void) {
   float polje[BR_REDAKA][BR_STUPACA] = {{[5] = 0.9f},
                                         {0.f},
                                         {31.1f, 32.2f, 33.3f, 34.2f},
                                         {1.f, 4.f, [6] = 7.7}};
   int i, j;
   for (i = 0; i < BR_REDAKA; i = i + 1) {
      for (j = 0; j < BR_STUPACA; j = j + 1) {
         printf("%5.1f", polje[i][j]);
      }
      printf("\n");
   }

   return 0;
}
```

2.
```c
#include <stdio.h>

#define BR_SLOJEVA 4
#define BR_REDAKA 3
#define BR_STUPACA 6

int main(void) {
   int polje[BR_SLOJEVA][BR_REDAKA][BR_STUPACA];
   int i, j, k;

   // u ovom zadatku lakse je polje napuniti petljama nego inicijalizatorom
   for (i = 0; i < BR_SLOJEVA; i = i + 1) {
      for (j = 0; j < BR_REDAKA; j = j + 1) {
         for (k = 0; k < BR_STUPACA; k = k + 1) {
            polje[i][j][k] = (i + 1) * 100 + (j + 1) * 10 + k + 1;
         }
      }
   }

   for (i = 0; i < BR_SLOJEVA; i = i + 1) {
      printf("%d. sloj\n", i + 1);
      for (j = 0; j < BR_REDAKA; j = j + 1) {
         for (k = 0; k < BR_STUPACA; k = k + 1) {
            printf("%5d", polje[i][j][k]);
         }
         printf("\n");
      }
      if (i < BR_SLOJEVA - 1) {
         printf("\n");
      }
   }

   return 0;
}
```

3.
```c
#include <stdio.h>

int main(void) {
   int m, n;

   printf("Upisite m, n > ");
   scanf("%d %d", &m, &n);

   int polje[m][n];

   printf("Upisite %d x %d clanova >\n", m, n);
   int i, j;
   for (i = 0; i < m; i = i + 1) {
      for (j = 0; j < n; j = j + 1) {
         scanf("%d", &polje[i][j]);
      }
   }

   int najmanji;
   for (j = 0; j < n; j = j + 1) {
      printf("%d. stupac: ", j + 1);
      najmanji = polje[0][j];
      for (i = 1; i < m; i = i + 1) {
         if (polje[i][j] < najmanji) {
            najmanji = polje[i][j];
         }
      }
      int prviPar = 1;
      for (i = 0; i < m; i = i + 1) {
         if (polje[i][j] == najmanji) {
            if (prviPar == 1) {
               prviPar = 0;
            } else {
               printf(", ");
            }
            printf("(%d, %d)", i, j);
         }
      }
      printf("\n");
   }

   return 0;
}
```

4.
```c
#include <stdio.h>

int main(void) {
   int m, n;

   printf("Upisite m, n > ");
   scanf("%d %d", &m, &n);

   int polje[m][n];

   printf("Upisite %d x %d clanova >\n", m, n);
   int i, j;
   for (i = 0; i < m; i = i + 1) {
      for (j = 0; j < n; j = j + 1) {
         scanf("%d", &polje[i][j]);
      }
   }

   for (j = 0; j < n; j = j + 1) {
      // u svakom stupcu j obavljaj sljedece
      int pomocni = polje[0][j]; // spasi j-ti clan iz 0-tog retka

      for (i = 0; i < m - 1; i = i + 1) {
         // clan u i-tom retku postavi na clan iz (i+1)-vog retka
         polje[i][j] = polje[i + 1][j];
      }

      // u zadnji redak stavi spaseni j-ti clan iz 0-tog retka
      polje[m - 1][j] = pomocni;
   }

   printf("Nakon posmaka:\n");
   for (i = 0; i < m; i = i + 1) {
      for (j = 0; j < n; j = j + 1) {
         printf("%4d", polje[i][j]);
      }
      printf("\n");
   }

   return 0;
}
```

5.
```c
#include <stdio.h>

#define RED 11

int main(void) {
   int i, j;
   int matrica[RED][RED];

   for (i = 0; i < RED; i = i + 1) {
      for (j = 0; j < RED; j = j + 1) {
         if (i == j || i == RED - 1 - j || j == 0 || i == 0 || j == RED - 1 ||
             i == RED - 1) {
            matrica[i][j] = 1;
         } else {
            matrica[i][j] = 8;
         }
      }
   }

   printf("Generirana kvadratna matrica:\n");
   for (i = 0; i < RED; i = i + 1) {
      for (j = 0; j < RED; j = j + 1) {
         printf("%2d", matrica[i][j]);
      }
      printf("\n");
   }

   return 0;
}
```

6.
```c
#include <stdio.h>

#define MIN_RED 4
#define MAKS_RED 8
#define MIN_STUP 5
#define MAKS_STUP 10

int main(void) {
   int m, n;

   do {
      printf("Upisite m [%d, %d] > ", MIN_RED, MAKS_RED);
      scanf("%d", &m);
   } while (m < MIN_RED || m > MAKS_RED);

   do {
      printf("Upisite n [%d, %d] > ", MIN_STUP, MAKS_STUP);
      scanf("%d", &n);
   } while (n < MIN_STUP || n > MAKS_STUP);

   int polje[m][n];

   printf("Upisite %d x %d clanova >\n", m, n);
   int i, j;
   for (i = 0; i < m; i = i + 1) {
      for (j = 0; j < n; j = j + 1) {
         scanf("%d", &polje[i][j]);
      }
   }

   int sumaPoStupcima[n];
   for (j = 0; j < n; j = j + 1) {
      sumaPoStupcima[j] = 0;
      for (i = 0; i < m; i = i + 1) {
         sumaPoStupcima[j] = sumaPoStupcima[j] + polje[i][j];
      }
   }

   printf("Sume po stupcima:\n");
   for (j = 0; j < n; j = j + 1) {
      printf("%5d", sumaPoStupcima[j]);
   }

   return 0;
}
```

7.
```c
#include <stdio.h>

int main(void) {
   int m, n;

   printf("Upisite m, n > ");
   scanf("%d %d", &m, &n);

   int polje[m][n];

   printf("Upisite %d x %d clanova >\n", m, n);
   int i, j;
   for (i = 0; i < m; i = i + 1) {
      for (j = 0; j < n; j = j + 1) {
         scanf("%d", &polje[i][j]);
      }
   }

   for (j = 0; j < n; j = j + 1) {
      // u svakom stupcu sortiraj clanove u retcima 0 do m-1
      for (i = 0; i < m - 1; i = i + 1) {
         int k, ind_min = i + 1;
         for (k = i + 2; k < m; k = k + 1) {
            if (polje[k][j] < polje[ind_min][j])
               ind_min = k;
         }
         if (polje[ind_min][j] < polje[i][j]) {
            int pomocna = polje[i][j];
            polje[i][j] = polje[ind_min][j];
            polje[ind_min][j] = pomocna;
         }
      }
   }

   printf("Nakon sortiranja stupaca:\n");
   for (i = 0; i < m; i = i + 1) {
      for (j = 0; j < n; j = j + 1) {
         printf("%4d", polje[i][j]);
      }
      printf("\n");
   }

   return 0;
}
```

8.
```c
#include <stdio.h>

int main(void) {
   struct pretinac_s {
      int sifra;
      int sirina;
      int visina;
      int dubina;
      float volumen;
   };

   int n;
   printf("Upisite broj pretinaca > ");
   scanf("%d", &n);

   struct pretinac_s pretinci[n];

   printf("Upisite podatke za pretince (%d)\n", n);
   int i;
   for (i = 0; i < n; i = i + 1) {
      printf("%2d. pretinac > ", i + 1);
      scanf("%d %d %d %d", &pretinci[i].sifra, &pretinci[i].sirina,
            &pretinci[i].visina, &pretinci[i].dubina);
      pretinci[i].volumen = (float)pretinci[i].sirina * pretinci[i].visina *
                            pretinci[i].dubina / 1000;
   }

   int ind_max, j;
   struct pretinac_s pomocni;
   for (i = 0; i < n - 1; i = i + 1) {
      ind_max = i + 1;
      for (j = i + 2; j < n; j = j + 1) {
         if (pretinci[j].volumen > pretinci[ind_max].volumen)
            ind_max = j;
      }
      if (pretinci[ind_max].volumen > pretinci[i].volumen) {
         pomocni = pretinci[i];
         pretinci[i] = pretinci[ind_max];
         pretinci[ind_max] = pomocni;
      }
   }

   printf("Sortirani pretinci:\n");
   for (i = 0; i < n; i = i + 1) {
      printf("%d = %6.2f litara\n", pretinci[i].sifra, pretinci[i].volumen);
   }

   return 0;
}
```