Pamięć dzielona

1. (1. pkt) Napisać program który tworzy w pamięci dzielonej (posix) macierz kwadratową *int *macA;* o rozmiarze *int N=12;* oraz wypełnia pola o współrzędnych x,y wartościami x*y. Po wykonaniu operacji pamięć należy odłączyć. Do wykorzystania są między innymi funkcje takie jak: *shm_open, ftruncate, mmap*. Macierz powinna wyglądać mniej więcej tak:

```
[  0][  0][  0][  0][  0][  0][  0][  0][  0][  0][  0][  0]
[  0][  1][  2][  3][  4][  5][  6][  7][  8][  9][ 10][ 11]
[  0][  2][  4][  6][  8][ 10][ 12][ 14][ 16][ 18][ 20][ 22]
[  0][  3][  6][  9][ 12][ 15][ 18][ 21][ 24][ 27][ 30][ 33]
[  0][  4][  8][ 12][ 16][ 20][ 24][ 28][ 32][ 36][ 40][ 44]
[  0][  5][ 10][ 15][ 20][ 25][ 30][ 35][ 40][ 45][ 50][ 55]
[  0][  6][ 12][ 18][ 24][ 30][ 36][ 42][ 48][ 54][ 60][ 66]
[  0][  7][ 14][ 21][ 28][ 35][ 42][ 49][ 56][ 63][ 70][ 77]
[  0][  8][ 16][ 24][ 32][ 40][ 48][ 56][ 64][ 72][ 80][ 88]
[  0][  9][ 18][ 27][ 36][ 45][ 54][ 63][ 72][ 81][ 90][ 99]
[  0][ 10][ 20][ 30][ 40][ 50][ 60][ 70][ 80][ 90][100][110]
[  0][ 11][ 22][ 33][ 44][ 55][ 66][ 77][ 88][ 99][110][121]
```

2. (1pkt) Napisać drugi programik, który podłączy się do wcześniej utworzonej pamięci i wypisze dowolny fragment macierzy. Do tego celu stworzyć osobną funkcję której parametry odpowiednio to wskaźnik do początku pamięci macA rozmiar boku pamięci macA współrzędne początku podmacierzy oraz rozmiar podmacierzy. Formatowanie na 3 znakach w funkcji *printf* to *%3d*. Przykładowo wywołania:

   wyswietl_podmacierz(macA,N,3,3,4);
   wyswietl_podmacierz(macA,N,N-2,N-2,2);
   Powinny dać nam wynik:

```
[  9][ 12][ 15][ 18]
[ 12][ 16][ 20][ 24]
[ 15][ 20][ 25][ 30]
[ 18][ 24][ 30][ 36]
[100][110]
[110][121]
```

3. (3pkt.) Stworzyć nowy program a w nim

- (1pkt)Napisać funkcję, która ma 4 parametry 3 wskaźniki do adresów macierzy oraz nazwę pliku. Funkcja ta powinna odczytać podany plik, na podstawie pierwszego wiersza ustalić rozmiar macierzy, stworzyć 3 macierze *macA*, *macB* i *macC* analogicznie jak w pierwszym punkcie, wczytać dane z podanego pliku do macierzy A i B (C wyzerować)i zwrócić rozmiar macierzy. Można użyć funkcji wyświetlającej (z punktu 2) by sprawdzić czy wszystko poszło jak należy.
  Poniższe wywołanie:

```
int N = 0; //rozmiar całej macierzy będzie odczytany
int *macA=NULL;
int *macB=NULL;
int *macC=NULL;
N=wczytaj(&macA,&macB,&macC,"mac.txt");
wyswietl_podmacierz(macA,N,0,0,N);
wyswietl_podmacierz(macB,N,0,0,N);
wyswietl_podmacierz(macC,N,0,0,N);
```

  powinno zwrócić następujący wynik:

```
[  0][  1][  2][  3]
[  1][  4][  1][  2]
[  0][  2][  2][  1]
[  1][  2][  1][  2]
[  0][  1][  2][  0]
[  2][  3][  1][  2]
[  1][  1][  2][  3]
[  0][  1][  0][  1]
[  0][  0][  0][  0]
[  0][  0][  0][  0]
[  0][  0][  0][  0]
[  0][  0][  0][  0]
```

- (1,5pkt.) Napisać funkcję, która dostaje za pomocą poniższej struktury adres początku podmacierzy A, B i C ich rozmiar, mnoży podmacierz AxB wynik zapisuje do podmacierzy C.
  Wywołanie:

```
Arg p;
p.rozmiar_glowny=4;
p.rozmiar_podm=2;
p.podmacA = macA;
p.podmacB = macB;
```

```
p.podmacC = macC;
mnoz_podmacierz_param(&p);
wyswietl_podmacierz(p.podmacC,N,0,0,4);
```

powinno wypisać na ekranie wymnożony fragment macierzy:

```
[  2][  3][  0][  0]
[  8][ 13][  0][  0]
[  0][  0][  0][  0]
[  0][  0][  0][  0]
```

W przypadku gdy liczymy całą macierz

```
p.rozmiar_podm=4;
```

powinniśmy dostać to co w przykładzie na wykładzie:

```
[  4][  8][  5][ 11]
[  9][ 16][  8][ 13]
[  6][  9][  6][ 11]
[  5][ 10][  6][  9]
```