Laboratorium 7. Cel: Zastosowanie tranzytywnego domknięcia do partycjonowania czasu (patrz wykład 12), 2 godziny.

Zadania:

1. Dla wskazanej pętli za pomocą kalkulatora ISCC znaleźć relację zależności, R, oraz przestrzeń iteracji, LD.
2. Zrobić rysunek pokazujący zależności w przestrzeni 6 x 6. W tym celu trzeba zastosować operator scan (R*[n]->{:n=6}); który wygeneruje wszystkie zależności w przestrzeni 6 x 6, pierwsza krotka wskazuje początek zależności (strzałki), druga krotka – koniec zależności (strzałki).
   !!!Uwaga: dla niektórych pętli w przestrzeni 6x6 zależności mogą nie istnieć, w takim przypadku należy rozszerzyć przestrzeń do rozmiaru 12x12.
3. Obliczyć tranzytywne domknięcie relacji R, R^+
4. Obliczyć relację R^k.
5. Znaleźć zbiór UDS zawierający początki krańcowe.
6. Obliczyć zbiór S(k):= R^k(UDS) - (R^+ . R^k)(UDS).
7. Utworzyć relację CODE.
8. Stosując relację CODE za pomocą operatora *scan* znaleźć wszystkie partycje czasu dla przestrzeni 6x6 i nanieść uzyskane partycje na rysunku utworzonym w p.2.
9. Wygenerować pseudokod i przekonwertować go na kod kompilowany.
10. Obliczyć zbiór, IND, zawierający niezależne iteracje pętli.
11. Jeśli zbiór IND nie jest pusty, to wygenerować pseudokod i kod kompilowany.
12. Zastosować program porównujący wyniki obliczeń (zadanie 7, L2) do sprawdzania poprawności kodu docelowego w przestrzeni 6x6.
13. Opracować sprawozdanie.

Patrz skrypt „L7” pokazujący dla przykładowej pętli realizację poszczególnych kroków zadań wyżej.

Warianty pętli:

1.
```
for(i=4;i<=n;i++)
  for(j=4;j<=n;j++)
    a[i][j] = a[i][j-1];
```
2.
```
for(i=1;i<=n;i++)
  for(j=2;j<=n;j++)
    a[i][j] = a[i][j-2];
```
3.
```
for(i=1;i<=n;i++)
  for(j=3;j<=n;j++)
    a[i][j] = a[i][j-3];
```
4.
```
for(i=1;i<=n;i++)
  for(j=1;j<=n;j++)
```

```
    a[i][j] = a[i-1][j-1];
```

5.

```
for(i=2;i<=n;i++)
  for(j=1;j<=n;j++)
    a[i][j] = a[i-2][j-1];
```

6.

```
for(i=2;i<=n;i++)
  for(j=2;j<=n;j++)
    a[i][j] = a[i-2][j-2];
```

7.

```
for(i=2;i<=n;i++)
  for(j=2;j<=n;j++)
    a[i][j] = a[i-2][j+2];
```

8.

```
for(i=1;i<=n;i++)
  for(j=0;j<=n;j++)
    a[i][j] = a[i-1][j+2];
```

9.

```
for(i=1;i<=n;i++)
  for(j=0;j<=n;j++)
    a[i][j] = a[i-1][j+1];
```

10.

```
for(i=1;i<=n;i++)
  for(j=0;j<=n;j++)
    a[i][j] = a[i+3][j+4];
```

11.

```
for(i=1;i<=n;i++)
  for(j=4;j<=n;j++)
    a[i][j] = a[i+3][j-4];
```

12.

```
for(i=1;i<=n;i++)
  for(j=4;j<=n;j++)
    a[i][j] = a[i+4][j-4];
```

13.

```
for(i=1;i<=n;i++)
  for(j=4;j<=n;j++)
    a[i][j] = a[i+5][j-4];
```

Sprawozdanie powinno zawierać: pętlę, skrypt implementujący zadania oraz wyniki wszystkich zadań.