J.NAWROCKI, M. ANTCZAK, H. ĆWIEK, W. FROHMBERG, A. HOFFA, M. KIERZYNKA, S. WĄSIK

POWTÓRKA

ZAD. 1. Narysować schemat blokowy dla problemu wyznaczania sumy cyfr dowolnej liczby naturalnej *n*
Parametry wejściowe: *n* – dowolna liczba naturalna.
Parametr wyjściowy: *Sum(n)*.
Podpowiedź: Wykorzystać operatory części całkowitej *(div)* oraz reszty z dzielenia *(mod)* przy dzieleniu wejściowej liczby naturalnej (n) i liczby *„10”*.
Przykład:
n=12 => Sum(n)=1+2=3.

ZAD. 2. Pamięć wirtualna składa się z 128 komórek o adresach logicznych od 0 do 127. Strona pamięci wirtualnej zawiera 2^3 komórek. Pamięć operacyjna ma 64 komórki o adresach fizycznych od 0 do 63 i jest podzielona na 8 bloków. Stan pamięci operacyjnej reprezentuje poniższy rysunek:

	Pam.oper.
0	**1**
...	**...**
6	**92**
...	**...**
19	**64**
...	**...**
40	**79**
...	**...**
62	**48**
63	**5**

a) w pamięci wirtualnej znajduje się ..... stron.

b) narysuj i wypełnij wartościami tablicę stron na podstawie zawartości pamięci operacyjnej oraz poniżej zdefiniowanych zależności:

ladrs=109 => fadrs=21, ladrs=99 => fadrs=43, ladrs=32 => fadrs=32, ladrs=28 => fadrs=4, ladrs=81 => fadrs=49, ladrs=15 => fadrs=31.

c) czy komórki znajdujące się pod następującymi adresami logicznymi są w PAO – 12 - ....., 72 – .....

ZAD. 3. Napisz program w języku C wyznaczający wartość średniej kwadratowej.

$$srednia = \sqrt{\frac{a_1^2 + ... + a_n^2}{n}}$$

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

double srednia(int n, int wart[]) {
        …
}

void main() {
        int n=0;
        int wart[255];
        printf("Podaj n=");
        scanf("%d",&n);
        …
        printf("Srednia kwadratowa to = %f\n", srednia(n,wart));
        system("PAUSE");
}
```

(*) gwiazdką oznaczone są zadania, które nie są realizowane na ćwiczeniach i są przeznaczone do wykonania jako zadania domowe.

ZAD. 4. Dla podanej poniżej tablicy prawdy skonstruuj tablicę Karnaugh, wyznacz dla niej funkcję (dla **1**) oraz przedstaw ją za pomocą funktorów logicznych.

a	b	c	d	e	y
0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	1	0	0
0	0	0	1	1	0
0	0	1	0	0	0
0	0	1	0	1	1
0	0	1	1	0	0
0	0	1	1	1	0
0	1	0	0	0	1
0	1	0	0	1	1
0	1	0	1	0	1
0	1	0	1	1	1
0	1	1	0	0	1
0	1	1	0	1	1
0	1	1	1	0	0
0	1	1	1	1	0
1	0	0	0	0	1
1	0	0	0	1	0
1	0	0	1	0	1
1	0	0	1	1	0
1	0	1	0	0	1
1	0	1	0	1	0
1	0	1	1	0	0
1	0	1	1	1	1
1	1	0	0	0	1
1	1	0	0	1	0
1	1	0	1	0	1
1	1	0	1	1	1
1	1	1	0	0	1
1	1	1	0	1	1
1	1	1	1	0	0
1	1	1	1	1	0

ZAD. 5. Napisz program obliczania **wartości a^n**, gdzie *a* i *n* są liczbami naturalnymi (a>0, *n >= 0*). Skorzystaj z instrukcji

```
imul s
```

która przesyła do pary rejestrów DX:AX wynik mnożenia liczby znajdującej się w rejestrze AX przez liczbę znajdującą się w rejestrze (lub innym miejscu) określonym przez operand *s*. Jeżeli wynik mieści się na 16 bitach i jest **nieujemny**, to wartość rejestru DX jest równa 0 (jeśli wynik mieści się na 16 bitach i jest **ujemny**, to wartością rejestru jest FFFF).
Na przykład instrukcja

```
imul bx
```

może być opisana następującą instrukcją w języku C:

```
ax = ax * bx;
```

Oto przykładowe dane testowe:

Wejście: n	**Wynik: a^n**	**Wejście: a**
BX	**AX**	**CX**
3	8	2

ZAD. 6. Załóżmy, że pracujemy w firmie turystycznej, która organizuje autobusowe wycieczki zagraniczne. Do naszych zadań należy stworzenie raportu dla każdej wycieczki (np. do Grecji) zawierającego numer autobusu, w którym jest najmniej miejsc wolnych. W pliku wejściowym każdy wiersz zawiera numer autobusu przypisanego danej wycieczce, maksymalną ilość dostępnych miejsc w autobusie oraz ilość osób siedzących w tymże autobusie w dniu wyjazdu. Przykładowy plik wejściowy znajduje się poniżej.

(*) gwiazdką oznaczone są zadania, które nie są realizowane na ćwiczeniach i są przeznaczone do wykonania jako zadania domowe.

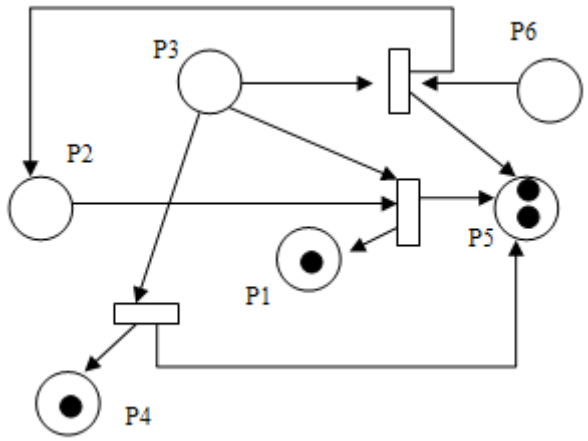
```
Autobus     L. miejsc   Zajęte
1           25          22
2           40          33
3           40          37
4           25          20
```

W rezultacie chcielibyśmy uzyskać plik wyjściowy, w którym wypisany byłby numer autobusu, który podczas wyjazdu miał najmniej wolnych miejsc. Dla powyższego pliku wejściowego powinniśmy uzyskać:

```
Najpelniejszy autobus to: 1
```

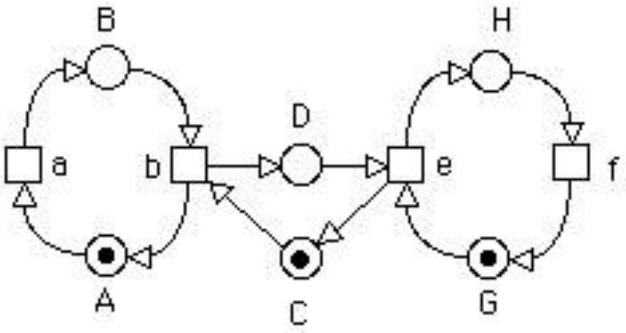
Napisz program, w AWK tak, aby spełniona była powyżej opisana zależność.

ZAD. 7. Poniższa sieć Petriego przedstawia pewien stan systemu w czasie **t**. Należy podać jeden z możliwych stanów systemu (miejsca, w których znajdowały się tokeny oraz liczbę tokenów dla danego miejsca) w czasie **t-1**, czyli stan przed odpaleniem tranzycji. Zakładamy, że wszystkie tranzycje, jeśli mogą, odpalają w tym samym czasie.



Nazwa miejsca	Liczba tokenów
P1	
P2	
P3	
P4	
P5	
P6	

ZAD. 8. Zmodyfikuj model sieci Petriego dla problemu producent – konsument dla magazynu o pojemności ***1***, w taki sposób, aby magazyn wyjściowego rozwiązania posiadał nieograniczoną pojemność (***n***).



ZAD. 9. Należy napisać program wyznaczania wartości *n-tego* wyrazu ciągu opisanego poniższą zależnością:

a) $a_n = \begin{cases} 1 & gdy \quad n = 0 \vee n = 1 \\ 2 * a_{n-1} + 3 * a_{n-2} \end{cases}$

b) $a_n = \begin{cases} 2 & gdy \quad n = 0 \vee n = 1 \\ 2 * a_{n-1} - a_{n-2} \end{cases}$

ZAD. 10. Skompletuj specyfikację wymagań do obsługi bankomatu korzystając z poniższej opowieści klienta:

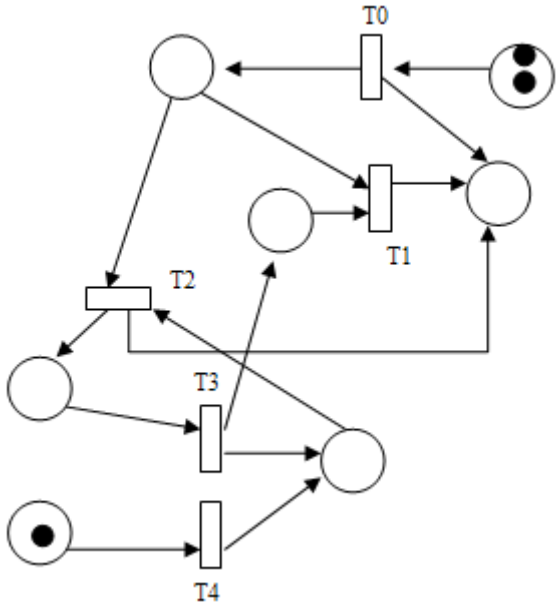
A) przed rozpoczęciem korzystania z bankomatu klient musi zostać zautoryzowany. Następnie bankomat prezentuje klientowi możliwe opcje (tylko jeżeli autoryzacja się powiodła). Klient może podjąć decyzję o pobraniu gotówki lub sprawdzeniu salda. W przypadku sprawdzenia salda bankomat prezentuje stan konta. Po potwierdzeniu odczytania salda bankomat ponownie prezentuje klientowi dostępne opcje.

ZAD. 11. Uzupełnij specyfikację wymagań z zadania 10 o następujące diagramy:

a) diagram przypadków użycia,
b) diagram sekwencji.

(*) gwiazdką oznaczone są zadania, które nie są realizowane na ćwiczeniach i są przeznaczone do wykonania jako zadania domowe.

ZAD. 12. Podaj, które z poniższych sekwencji odpaleń przejść są możliwe dla sieci Petriego zaprezentowanej poniżej.



1. T4, T0, T2, T3, T1
2. T0, T4, T2, T3, T0, T1
3. T4, T0, T2, T3
4. T0, T0, T4, T2, T3, T1
5. T0, T2, T4, T2, T3
6. T0, T0, T4, T2, T3, T2, T3

Odp.: ……

ZAD. 13. Należy zaproponować gramatykę dla języka opisanego za pomocą następującego wyrażenia regularnego $1^n 2^m 3^+ 2^m 1^n$, gdzie n,m>0 i wywieźć następujące zdania:
a) 12321
b) 1231

ZAD. 14. Wykorzystując rozszerzoną notację Backusa-Naura (EBNF) uogólnij przykład instrukcji języka C **while(((W&&W))||((W&&W))) {}**, gdzie **W** reprezentuje dowolne wyrażenie logiczne, które może zostać uproszczone do wartości logicznych True (**1**) i False (**0**).

ZAD. 15. Napisz w LEX-ie program który sprawdza czy ciąg znajdujący się w pliku wejściowym zawiera na zmianę A i B. Wykorzystaj makro BEGIN i dwa stany.

Plik wejściowy
ABABAB

BABAB
BBAA

Plik wyjściowy:
ABABAB T
T
BABAB T
BBAA F

ZAD. 16. Mając dany przedstawiony poniżej parser napisany w LEX-ie napisać w YACC-u program, który mając dany plik złożony z ciągu cyfr wypisze czy w pliku znajduje się więcej cyfr parzystych, czy nieparzystych.

Plik wejściowy:
017891

Plik wyjściowy:
Wiecej nieparzystych

LEX code:

```
%{
  #include "ytab.h"
%}
%%
[02468]          {yylval=atoi(yytext);return p;}
[13579]          {yylval=atoi(yytext);return n;}
\n               ;
.                {YY_FATAL("Unexpected character!!!");}
```

(*) gwiazdką oznaczone są zadania, które nie są realizowane na ćwiczeniach i są przeznaczone do wykonania jako zadania domowe.