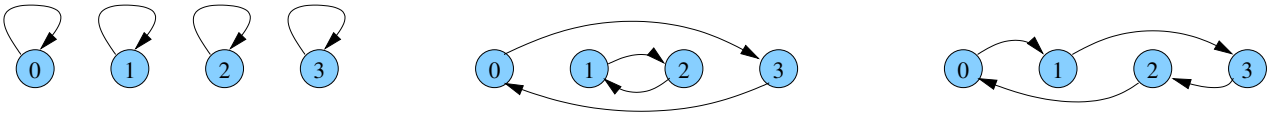
# Hľadanie cyklov v permutáciach

Permutáciou bude pre nás pole int-ov dĺžky $n$, v ktorom sa každé z čísel $\{0, 1, \ldots, n-1\}$ vyskytuje práve raz. Napríklad pre $n = 4$ príklady permutácií sú $a_1 = \{0, 1, 2, 3\}$, $a_2 = \{3, 2, 1, 0\}$, $a_3 = \{1, 3, 0, 2\}$ atď. Intuitívne si permutáciu $a$ môžeme predstaviť ako obrázok s bodkami očíslovanými $0, \ldots, n-1$ nakreslenými v rade a každý prvok $a[i]$ nakresliť ako šípku z $i$ do $a[i]$. Z každej bodky jedna šípka vychádza a jedna do nej vchádza. Na obrázku nižšie vidíme takto znázornené permutácie $a_1$, $a_2$ a $a_3$.




Cyklus dĺžky $k$ v permutácii $a$ je taká postupnosť navzájom rôznych čísel $i_1, i_2, \ldots, i_k$ z množiny $\{0, \ldots, n-1\}$, že $a[i_1] = i_2$, $a[i_2] = i_3$, $a[i_3] = i_4$, $\ldots$, $a[i_{k-1}] = i_k$ a $a[i_k] = i_1$. V našej obrázkovej predstave cyklus dostaneme, ak začneme na niektorej bodke a budeme sa pohybovať po šípkach, až kým neprídeme späť na bodku, na ktorej sme začali.

Napríklad v permutácii $a_1$ je každý index $0, \ldots, 3$ cyklom dĺžky 1. V permutácii $a_2$ máme dva cykly dĺžky 2: $(0, 3)$ a $(1, 2)$. V permutácii $a_3$ je jeden cyklus dĺžky 4: $(0, 1, 3, 2)$. Všimnite si, že tento cyklus by sme mohli zapísať aj tak, že začneme od iného prvku, napr. $(1, 3, 2, 0)$ alebo $(3, 2, 0, 1)$ alebo $(2, 0, 1, 3)$. Všetky takéto zápisy cyklu, ktoré sa líšia iba začiatkom, budeme považovať za ekvivalentné.

**Cvičný príklad 1:** Napíšte funkciu, ktorá dostane permutáciu $a$, jej dĺžku $n$ a index $i$ a vypíše cyklus permutácie obsahujúci prvok $i$.

**Riešenie:** Pomocou príkazu `i=a[i]` sa posunieme vždy na ďalší prvok cyklu. Opakujeme to dovtedy, kým sa nevrátime na štartovaciu pozíciu. Nižšie sú dva veľmi podobné spôsoby, ako to zapísať:

```
void printCycle(int a[], int n, int i) {
    int start = i;
    cout << " " << i;
    i = a[i];
    while(i != start) {
        cout << " " << i;
        i = a[i];
    }
    cout << endl;
}
```

```
void printCycle(int a[], int n, int i) {
    int start = i;
    while(true) {
        cout << " " << i;
        i = a[i];
        if (i == start) break;
    }
    cout << endl;
}
```

**Cvičný príklad 2:** Napíšte funkciu, ktorá dostane permutáciu $a$ a jej dĺžku $n$ a vypíše všetky cykly tejto permutácie, každý na jeden riadok. Dajte pozor, aby ste každý cyklus vypísali práve raz.

**Riešenie:** mohli by sme jednoducho zavolať `printCycle` pre všetky hodnoty $i$ od 0 po $n-1$, ale potom by sme každý cyklus vypísali toľkokrát, koľko má prvkov. Ak chceme mať každý cyklus vypísaný iba raz, môžeme si počas `printCycle` značiť, ktoré hodnoty sme už vypísali, napr. tak, že ich v poli $a$ prepíšeme hodnotou -1 (ktorá sa inak v permutácii nevyskytuje). Potom `printCycle` zavoláme len pre také $i$, kde $a[i]$ ešte nebolo zmenené na -1.

Toto riešenie zničí hodnoty v poli $a$. Ak by sme ich chceli zachovať, značky by sme si mohli robiť do pomocného poľa (stačilo by nám pole bool-ov).

```
void printCycle(int a[], int n, int i) {
    int start = i;
    while(true) {
        cout << " " << i;
        int next = a[i];  // odlozime si povodne a[i]
        a[i] = -1;        // poznacime si, ze a[i] je uz pouzite
        i = next;
        if (i == start) break;
    }
    cout << endl;
}

void printAllCycles(int a[], int n) {
    for (int i=0; i<n; i++) {
        if(a[i]>=0) printCycle(a, n, i);
    }
}
```

**Cvičný príklad 3:** Napíšte funkciu, ktorá dostane permutáciu $a$ a jej dĺžku $n$ a vráti počet cyklov v tejto permutácii.

Riešenie skúste napísať vy.