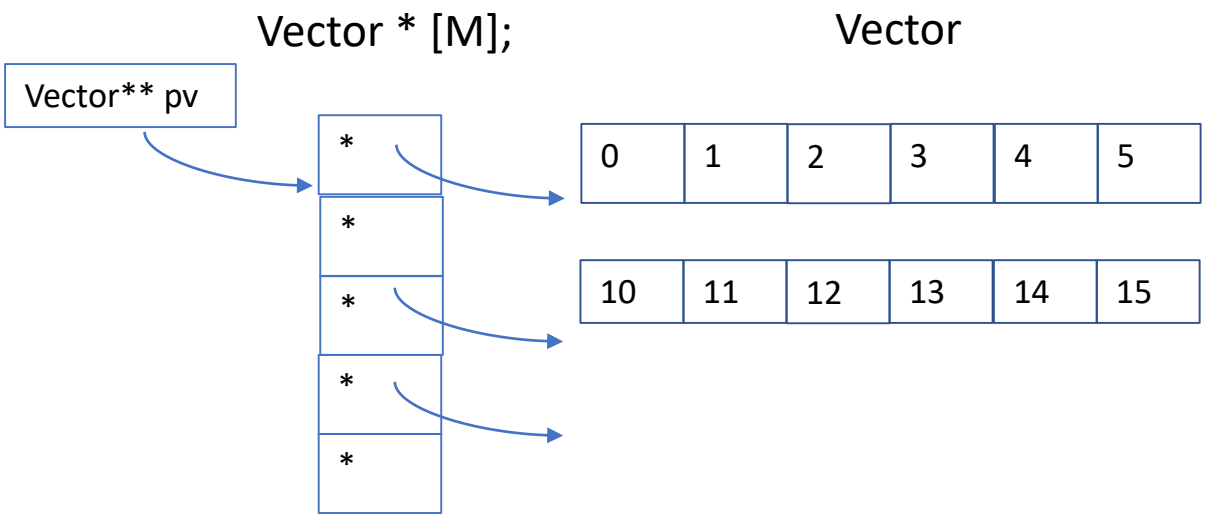
Supraîncărcare operator indexare şi operator functie

operator[]

- sens originar: localizare element într-o multime; **trebuie păstrat sensul**
- recunoscut implicit pentru vector de obiecte:
  - Pers vp[5]; // default cons
  - Pers vp[] = { Pers(“Unu”, 1000), Pers(“Doi”, 2000) }; // cons + copy cons
- **exemple de supraîncărcare:**
  - vector alocat dinamic
  - masive dinamice multidimensionale: **c[i][j][k]** = 123;
    // supraîncărcare în cascadă
  - extragerea bitului *k* dintr-o configuratie dată
  - implementare dicționar: (*key, value*); ex. căutarea rapidă a unei persoane după CNP, într-o listă de pointeri la persoane, sortată după CNP

```
Pers * Index::operator[ ](char *cnp)
{
        int poz;
        if(cautBin( cnp, poz) )
                return lista[poz]; // pointer la persoana cu CNP dat
        else return NULL;
}
```

- pentru **acces doar în read**, operatorul întoarce **valoare** sau ***const &*** / ***const** **



Clasa Matrix definită ca vector de linii Vector

Supraîncărcare operator indexare şi operator functie

operator() ( )

- sens originar: apel de functie **f(x)**
- sens la supraîncarcare:
  - un obiect ce transporta pointer de functie
  - în raport de context se transforma in apel de functie: o(x); o(x,y);
  - implementări curente pentru obiecte de tip ***comparator***

```
class MaiMic // clasa ce supraincarca operator functie
{
        public: bool operator()(double a, double b) { return a < b; }
};

void Vector::sort(MaiMic mm)
{
        for (int i = 0; i<dim - 1; i++)
                for (int j = i + 1; j<dim; j++)
                        if (!mm(pe[i], pe[j]))
                        {
                                double aux = pe[i]; pe[i] = pe[j]; pe[j] = aux;
                        }
}
```

- clasa poate contine mai multe supraîncărcări ale lui **operator()**
- operatorul nu are cardinalitate impusă
- recunoaștere dupa dubla pereche de paranteze: **() ()**