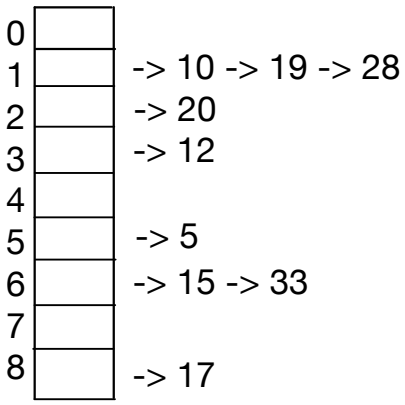
Seminar 5

1. **Dictionar ordonat cu reprezentare pe tabela de dispersie cu rezolvare coliziuni prin liste independente (separate chaining)**
2. **Dictionar cu reprezentare pe tabela de dispersie cu rezolvare coliziuni prin liste intrepatrunse (coalesced chaining)**

1. 9 chei : 5, 28, 19, 15, 20, 33, 12, 17, 10
   m = 9
   dispersie prin diviziune => d(c) = c % m



- mentinem listele independente ordonate dupa relatia R
- daca se parcurge DO cu iteratorul, ordinea asteptata este : 5, 10, 12, 15, 17, 19, 20, 28, 33
- interclasare liste independente pe rand

c	d(c)
5	5
28	1
19	1
15	6
20	2
33	6
12	3
17	8
10	1

- interclasare liste indep pe rand
lista1 cu lista2 -> lista12
lista12 cu lista 3 -> lista123
lista123 cu lista 4 -> lista1234
...

$2\ell + 3\ell + \dots + m\ell \approx \ell \cdot \frac{m(m+1)}{2} = \frac{n}{m} \cdot \frac{m(m+1)}{2} = \theta(m \cdot m)$

interclaseaza liste (d, l) → lista rezultata
este apelata din constructorul IteratorDictionar

NodT
   c : TCheie + v : TValoare
   urm : *TNod

DictionarOrdonat
   m : Intreg
   d : TFunctie : TCheie -> {0,...,m-1}
   R : Relatie : TCheie x TCheie -> {A,F}
   t : (*NodT)[ ]

IteratorDictionar
   d : DictionarOrdonat
   l : *NodT
   nodcrt : *NodT

Complexitati operatii iterator
- creeaza : : θ(n*m)
- valid : θ(1)
- urmator : θ(1)
- element : θ(1)

TD cu m pozitii
DO cu n elemente
} lungime medie lista indepemndenta n/m (factor de incarcare) = ℓ

2. 7 chei : 5, 18, 16, 15, 13, 31, 26
m = 13
dispersie prin diviziune

	0	1	2	3	4	5	6	7	8	9	10	11	12
e	18	13	15	16		5							
urm	1	4	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1

c	d(c)
5	5
18	5
16	3
15	2
13	0
31	5
26	0

primLiber = ~~0 1~~ 4

	0	1	2	3	4	5	6	7	8	9	10	11	12
e	18	13	15	16	31	5	26						
urm	1	4	-1	-1	6	0	-1	-1	-1	-1	-1	-1	-1

c	d(c)
5	5
18	5
16	3
15	2
13	0
31	5
26	0

primLiber = ~~0 1 4 6~~ 7

TElement
c : TCheie
v : TValoare

Dictionar
m : Intreg
d : Functie : TCheie -> {0,...,m-1}
e : TElement[ ]
urm : Intreg[ ]
primLiber : Intreg

subalgoritm cauta(dict, c):
i <- dict.d(c)
cat timp (i != -1 si dict.e[i].c != c)
i <- dict.urm[i]
sf cat timp
daca i = -1 atunci
cauta <- NULL_TValoare
altfel
cauta <- dict.e[i].v
sf daca
sf subalg

	0	1	2	3	4	5	6	7	8	9	10	11	12
e	~~18~~	~~13~~	15	16	31	~~5~~	26						
urm	~~1~~	~~4~~	-1	-1	6	0	-1	-1	-1	-1	-1	-1	-1

18 -1

```
subalg sterge(dict, c):
i <- dict.d(c)
j <- -1
cat timp (i != -1 si dict.e[i].c != c)
    j <- i
    i <- dict.urm[i]
sf cat timp
daca i = -1
    @cheia nu exista
altfel
gata <- fals
repeta
    p <- dict.urm[i]
    pp <- i
    cat timp (p!=-1 si dict.d(dict.e[p]) != i)
        pp <- p
        p <- dict.urm[p]
    sf cat timp
    daca p != -1 atunci
        dict.e[i] <- dict.e[p]
        i <- p
        j <- pp
    altfel
        gata <- adevarat
    pana cand gata
sf daca
k <- 0
cat timp (k < dictm si j = -1)
    daca dict.urm[k] = i
        j <- k
    altfel
        k <- k + 1
sf cat timp
daca j != -1 atunci
    dict.urm[j] <- dict.urm[i]
sf daca
dict.e[i] <- NULL_Element
dict.urm[i] <- -1
daca dict.primLiber > i atunci
    dict.primLiber <- i
sf daca
```