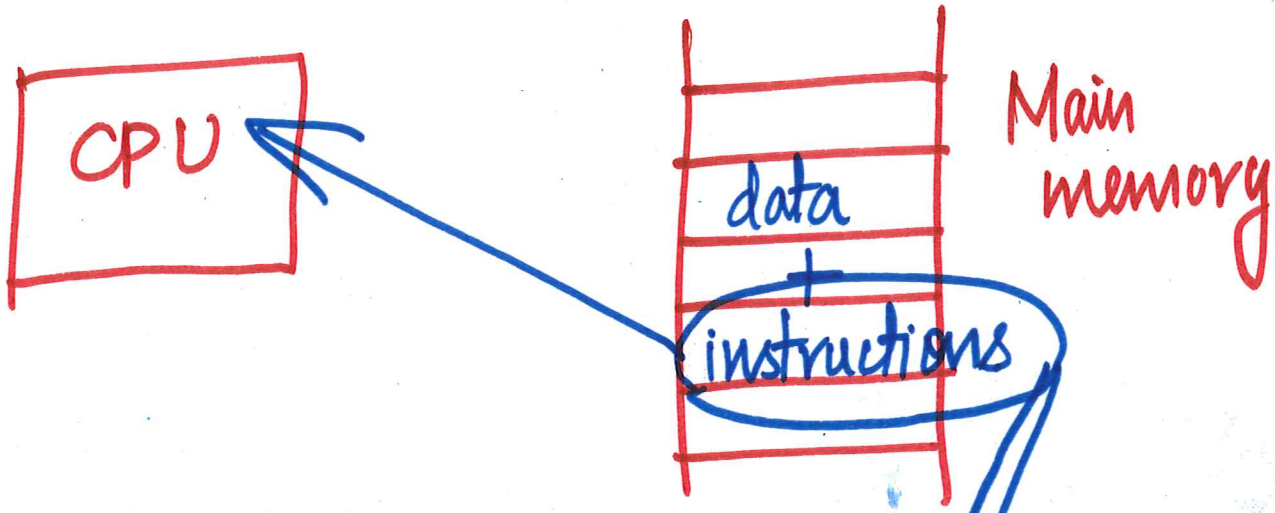
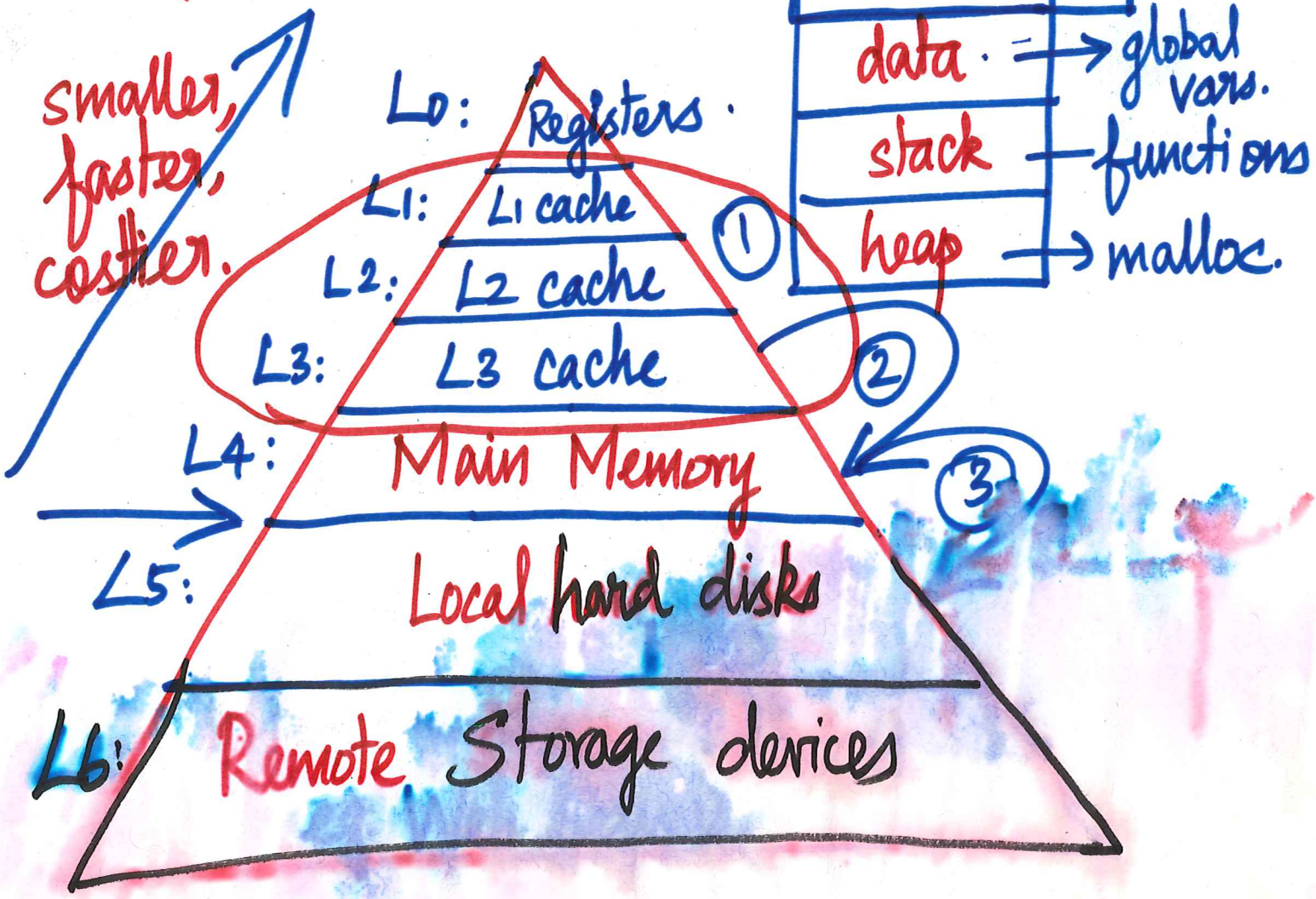
Memory



Memory Hierarchy

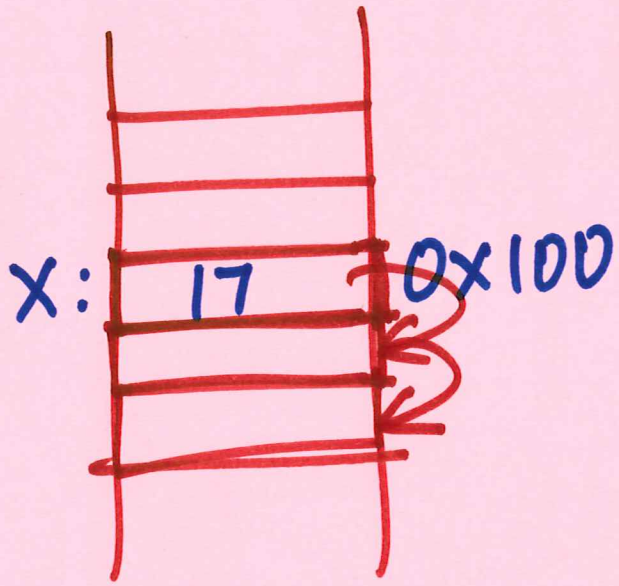


cache – 1 to 30 CPU cycles

main memory – 50 to 200 "

Hard disk – 1-10 million "

Locality



Access data items over and over

(or)

Access memory locations that are adjacent to each other.

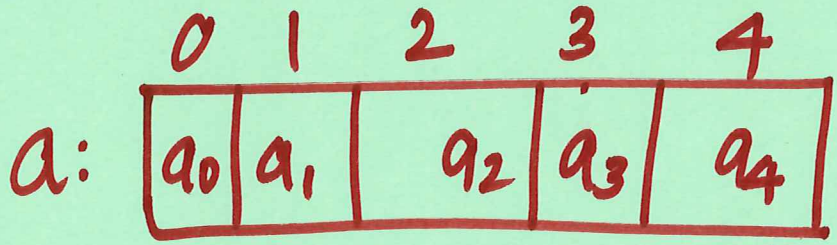
Locality

Temporal
"same mem. loc. accessed."

Spatial
nearby mem. locs. are accessed.

Why locality is important?

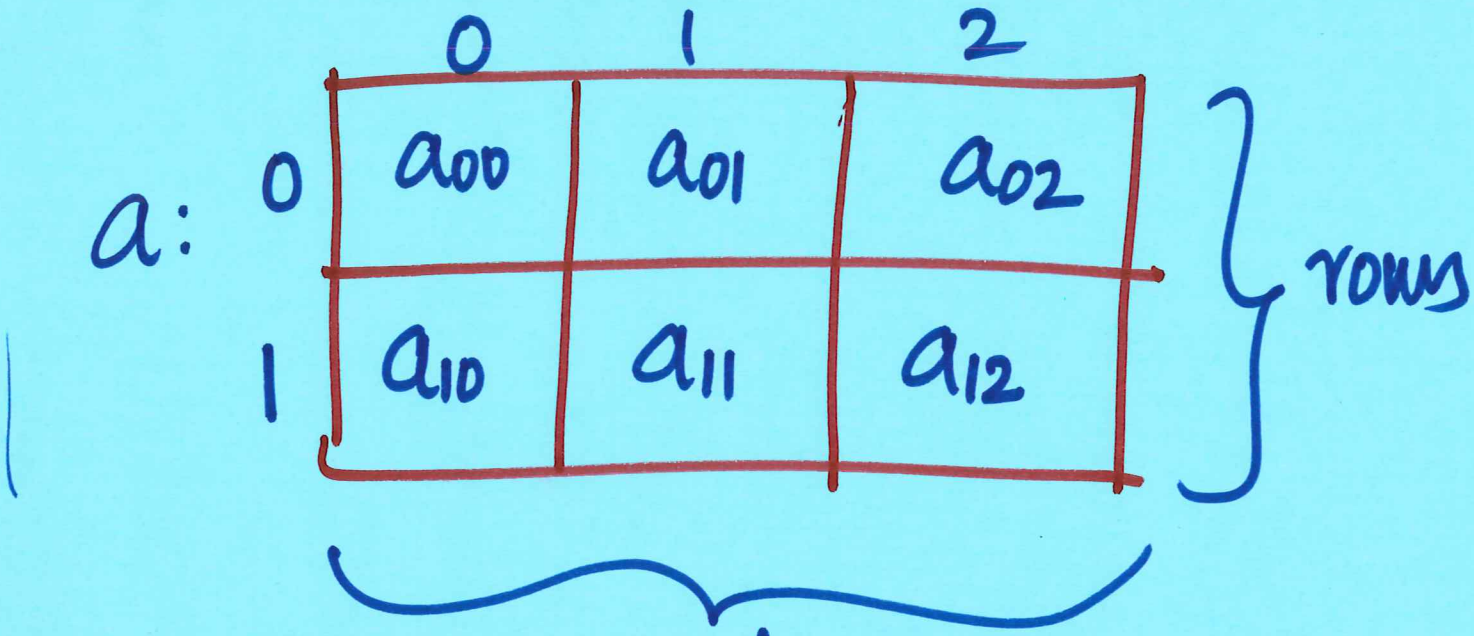
Programs with good locality
- run faster.



```
int sum_array(int a[], int n)
{
    int i, sum = 0;
    for(i=0; i<n; i++){
        sum += a[i];
    }
    return sum;
}
```

Array elem	a0	a1	a2	a3	a4
~~index~~					
Address	0	4	8	12	16
Access order	1	2	3	4	5

```
int sum_arr_rows(int a[][], int m,
                 int n)
{   int i, j, sum = 0;
    for(i=0; i<m; i++) {
        for(j=0; j<n; j++) {
            sum += a[i][j];
    }   }
    return sum;
}
```




```
int a[2][3];
```

elem:	a_{00}	a_{01}	a_{02}	a_{10}	a_{11}	a_{12}
Addr:	0	4	8	12	16	20
Access order. row wise	1	2	3	4	5	6
Access Order	1	3	5	2	4	6

sum_arr_cols

```
for(j=0; . . . )
    for(i=0; . . . )
        sum += a[i][j];
```

stride-1 reference pattern.

sequential " "

stride-K reference pattern

Memory access.

0 4 8 12 16 . . . .

stride-4 reference pattern.

$$K \propto \frac{1}{\text{locality}}$$