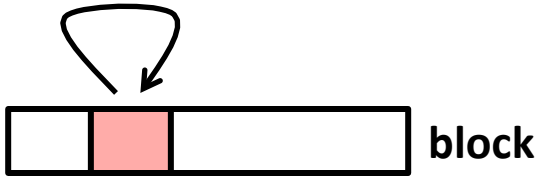
Section 7: Memory and Caches

- ~~Cache basics~~
- Principle of locality
- Memory hierarchies
- Cache organization
- Program optimizations that consider caches

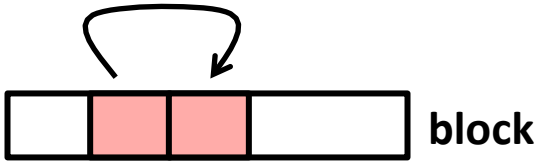
Why Caches Work

- **Locality: Programs tend to use data and instructions with addresses near or equal to those they have used recently**

- **Temporal locality:**
  - Recently referenced items are *likely* to be referenced again in the near future



- **Spatial locality:**
  - Items with nearby addresses *tend* to be referenced close together in time



  - How do caches take advantage of this?

Example: Locality?

```
sum = 0;
for (i = 0; i < n; i++)
        sum += a[i];
return sum;
```

Example: Locality?

```
sum = 0;
for (i = 0; i < n; i++)
        sum += a[i];
return sum;
```

- **Data:**
  - Temporal: **`sum`** referenced in each iteration
  - Spatial: array **`a[]`** accessed in stride-1 pattern

Example: Locality?

```
sum = 0;
for (i = 0; i < n; i++)
        sum += a[i];
return sum;
```

- **Data:**
  - Temporal: **`sum`** referenced in each iteration
  - Spatial: array **`a[]`** accessed in stride-1 pattern
- **Instructions:**
  - Temporal: cycle through loop repeatedly
  - Spatial: reference instructions in sequence

Example: Locality?

```
sum = 0;
for (i = 0; i < n; i++)
        sum += a[i];
return sum;
```

- **Data:**
  - Temporal: **`sum`** referenced in each iteration
  - Spatial: array **`a[]`** accessed in stride-1 pattern
- **Instructions:**
  - Temporal: cycle through loop repeatedly
  - Spatial: reference instructions in sequence

- **Being able to assess the locality of code is a crucial skill for a programmer**

Another Locality Example

```
int sum_array_3d(int a[M][N][N])
{
    int i, j, k, sum = 0;

    for (i = 0; i < N; i++)
        for (j = 0; j < N; j++)
            for (k = 0; k < M; k++)
                sum += a[k][i][j];
    return sum;
}
```

- What is wrong with this code?
- How can it be fixed?