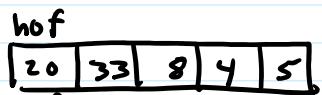


## Arrays and Pointers

### Stack

```
int hof[] = {20, 33, 8, 4, 5};
```

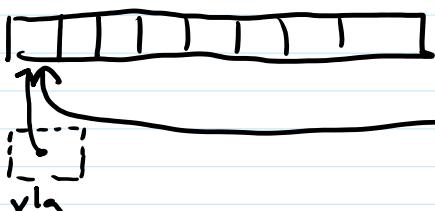


foo(hof)

~~hof = malloc(...)~~ illegal

hof[2]

```
int vla[size];
```



foo(vla)

~~vla = malloc(...)~~ still illegal

dynamic dynamic + 2

```
int *dynamic = malloc(sizeof(int) * size);
```

dynamic

malloc(...)

dynamic[2] ≡ \*(dynamic + 2)

### Heap

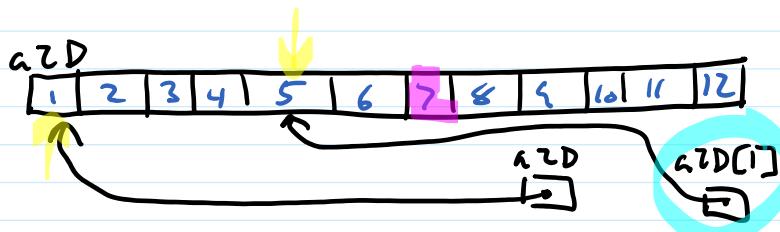
Diagram illustrating heap memory allocation and pointer arithmetic.

The heap is shown as a horizontal array of squares. Two green arrows point to the first two squares, labeled "dynamic" and "dynamic + 2".

A pointer box labeled "dynamic" points to the first square. A callout box labeled "malloc(...)" points to the second square.

A pointer box labeled "dynamic[2]" points to the third square, with a callout box labeled "\*(dynamic + 2)" pointing to it.

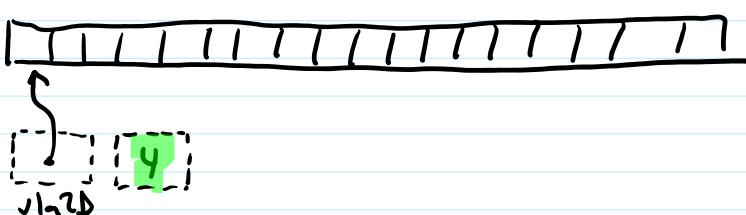
```
int a2D[3][4] = {{1, 2, 3, 4}, {5, 6, 7, 8}, {9, 10, 11, 12}};
```



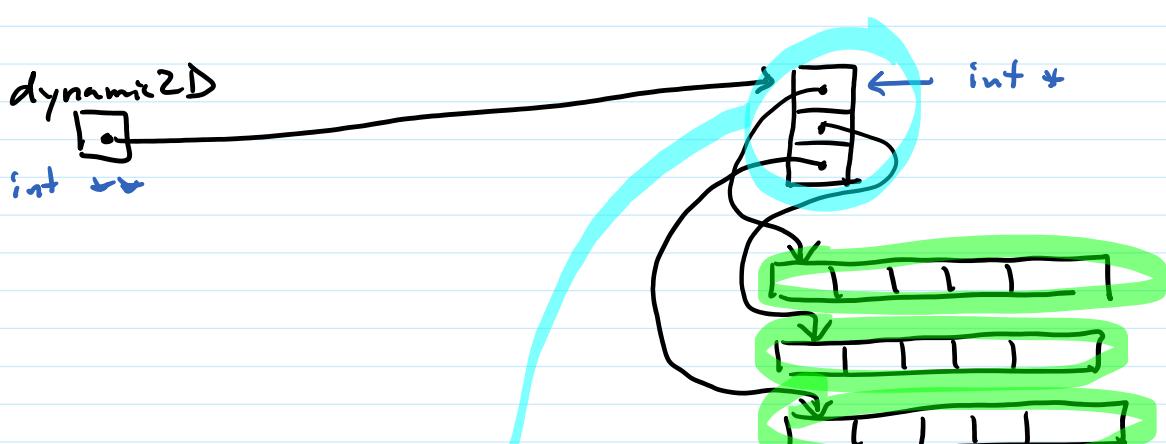
a2D is ptr to row

ptr to array of 4 ints  
a2D[1] is ptr to next row of 4 ints

```
int vla2D[size][size + 1];
```



size



```

int **dynamic2D;
dynamic2D = malloc(h * sizeof(int *));
for (int i = 0; i < h; i++)
{
    dynamic2D[i] = malloc(w * sizeof(int));
    for (int j = 0; j < w; j++) /* initialize ind elts */
}
...
for (int i = 0; i < h; i++)
{
    free(dynamic2D[i]);
}
free(dynamic2D);

```

free array of rows

free each row

can have different width for each row  
(until for triangular arrays)

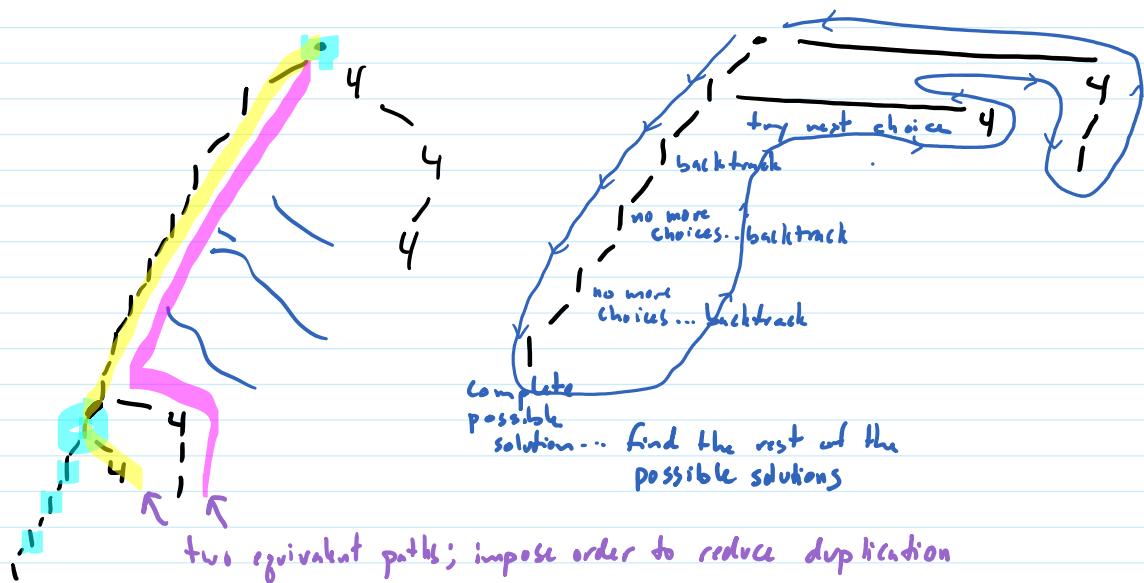
## Backtracking

**Problem:** given positive integer  $n$ , determine how to write  $n$  as the sum of as few squares as possible

**Examples:**  $5 = 1^2 + 2^2$

$$12 = 2^2 + 2^2 + 2^2$$

$$2018 = 13^2 + 43^2$$



recursion on partial solution:

if partial solution is  $1+1+1$ ,  
find best solution for  $1+1+1+1$   
find best solution for  $1+1+1+4$   
find best solution for  $1+1+1+9$

} for each next piece of solution  
add it to current partial solution  
recursively optimize the new partial solution  
(remove it from partial soln so we can add the next later)

Revisit this problem later in the semester? (dynamic programming)