1. $i1 = -5,$ $i2 = 66$ } swap ftn

       $i1 = 66,$ $i2 = -5$

2. program tracing

numbers:

1	1	2	4	8	0	0	0	0	0
0	1	2	3	4	5	6	7	8	9

j	i
0	0, 1 — —
1	0, 1
2	0, 1, 2 — — —
3	0, 1, 2 — —
4	0, 1, 2, 3 —
⋮	

ANS:

1 1 2 4 8 16 32 64 128 256

3. write a __program__

    #include <stdio.h>

```c
int main (void)
{
        float f, c;

    printf ("Enter temp (F): ___");
    scanf ("%f", &f);

    // convert fahrenheit to celsius
    c = (f - 32.0) / 1.8;
    printf ("Temp (C): ___ %.0f\n", c);

    return 0;
}
```

4. function

```c
int findmin (int A[], int size)
{
        int i, minVal = A[0];
```

```
for (i=0; i< size; i++)
    if ( A[i] < minVal ) minVal = A[i];

return minVal;
}
```

5. Calc. average:
```
float mean (float A[], int n)
{   int i;
    float sum = 0.0;

    for(i=0; i<n; i++)
        sum += A[i];

    sum /= (float)n;

    return (sum);
}
```
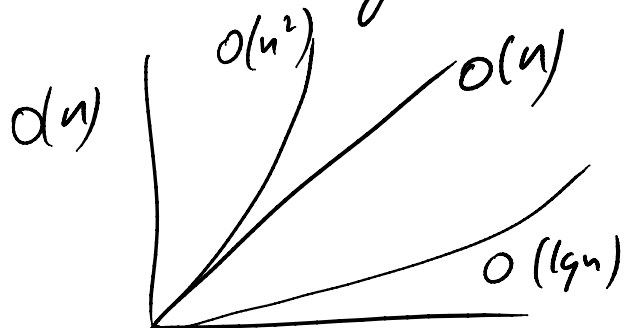
in 2IL
analysis of algorithm
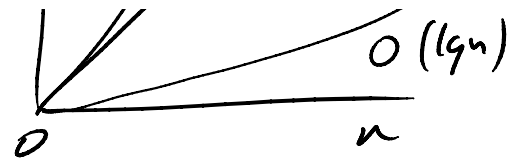given input size n
(i.e. list of n integers)

how long does the above
search alg. take?

how many operations?
or how many of the n
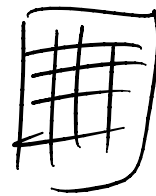items need to be inspected?

Ans: $O(n)$ — "big oh"

"on the order of"

$O(n^2)$ $O(n)$

$O(n)$ $O(\lg n)$

$O(\lg n)$

Is $O(n)$ good?

it's linear, not hard,

better than $O(n^2)$

not as good as $O(\lg n)$

for image of $n \times n$ size

this means $O(n^2)$

can improve on the GPU

$\rightarrow$ play-play to find min/max

$\Rightarrow O(\lg n)$
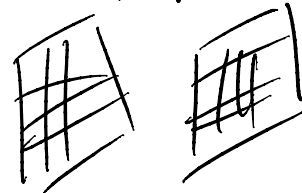
6. 7, $\begin{pmatrix} 4 \\ 5 \end{pmatrix}$, $\begin{pmatrix} -2 \\ 3 \\ 1 \end{pmatrix}$

7 Q − P

8. <u>Declare</u> <u>ptr to fun</u>

int (* pfun)(int, int); // declaration

9. double (* hits)(---)

↑

this hits "morphs" into
many (poly) instances ⇒ polymorphism

10. void list_add (list_t * list, void *data)
                              (link_t *)
   list_init { link_t *new = ^malloc(sizeof(link_t))

           new → next = NULL;
           new → data = data;

           if (list → head == NULL)
               list → head = new;

           else

```
else
        list -> tail -> next = new

list -> tail = new ;
}
```

lab 8

fleshing out date object

(see pervar notes)