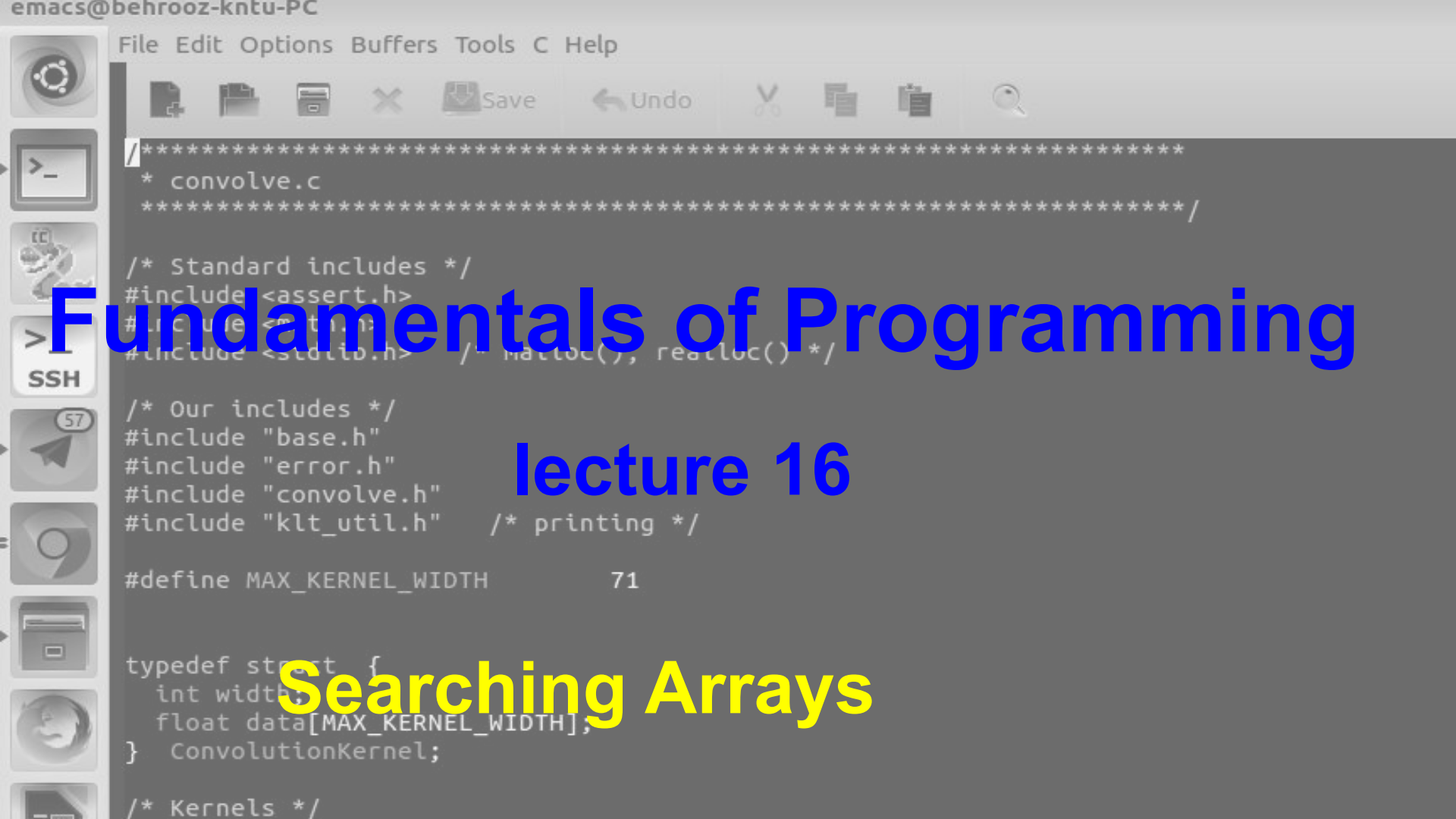

emacs@behrooz-kntu-PC
File Edit Options Buffers Tools C Help
Save
Undo
/**********
* convolve.c
**********/
/* Standard includes */
#include <assert.h>
/* Our includes */
#include "base.h"
#include "error.h"
#include "convolve.h"
#include "klt_util.h" /* printing */
#define MAX_KERNEL_WIDTH 71
typedef struct {
int width;
float data[MAX_KERNEL_WIDTH];
} ConvolutionKernel;
/* Kernels */
SSH
Fundamentals of Programming
lecture 16
Searching Arrays

Searching arrays

25	19	14	18	27	6	32	18	20	1	21

- Is the number 19 in the array?
- What is its index?

```
index = search(key, array, size);
```

linear search

```c
int linearsearch(int key, const int array[], int size) {
  for (int i = 0; i < size; i++)
    if (array[i] == key)
      return i;
  return -1;
}
```

linearsearch.c

linear search

```c
void printArray(const int array[], int n);
int linearsearch(int key, const int array[], int size);

int main() {
  int array[] = {25, 19, 14, 18, 27, 6, 32, 18, 20, 1, 21};
  int size = sizeof(array) / sizeof(array[0]);
  int key;

  printArray(array,size);

  while (1) {
    scanf("%d", &key);

    if (key == -1)
      break;

    int index = linearsearch(key, array, size);

    printf("%d\n", index);
  }

  return 0;
}


int linearsearch(int key, const int array[], int size) {
  for (int i = 0; i < size; i++)
    if (array[i] == key)
      return i;
  return -1;
}
```

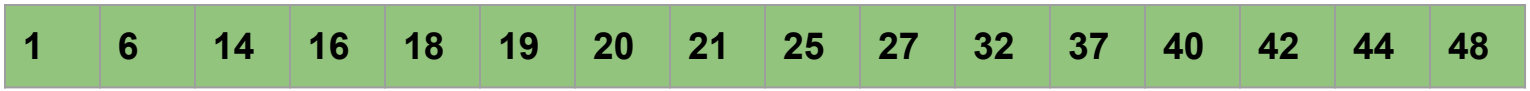
linearsearch.c

linear search

- Average no. of elements to examine
- What if the array is sorted?

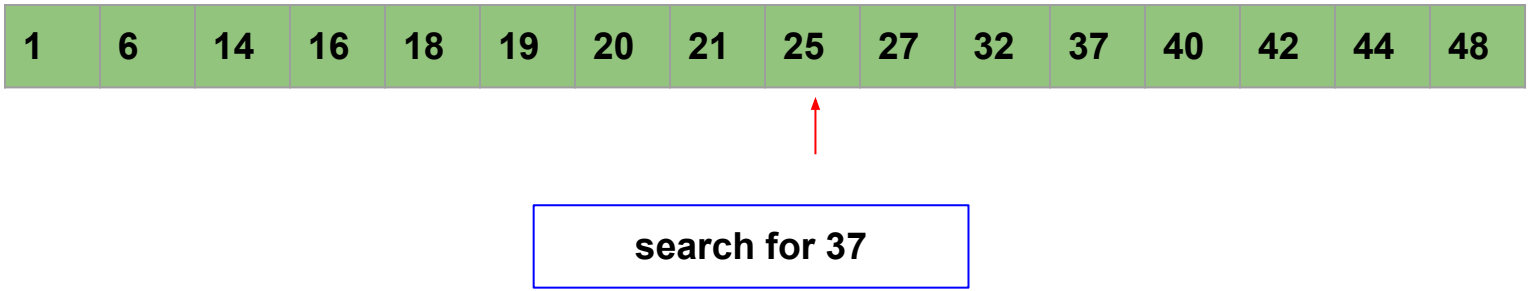
1	6	14	16	18	19	20	21	25	27	32

Binary search

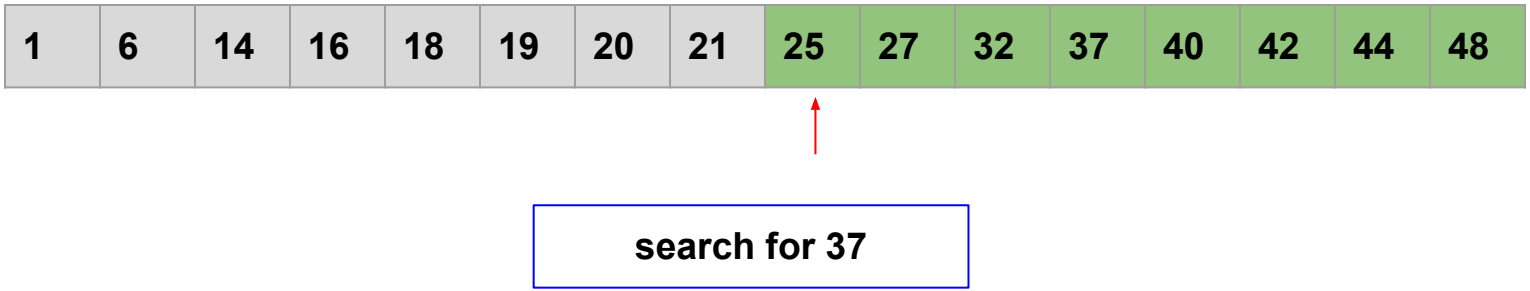


search for 37

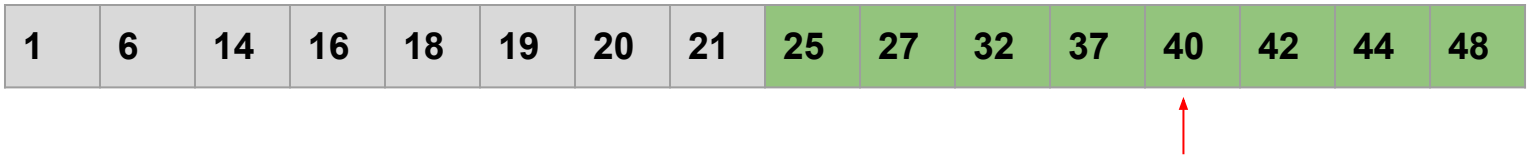
Binary search



Binary search

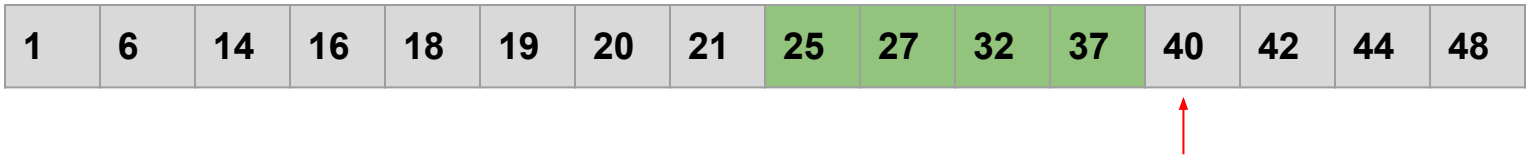


Binary search



search for 37

Binary search



search for 37

Binary search

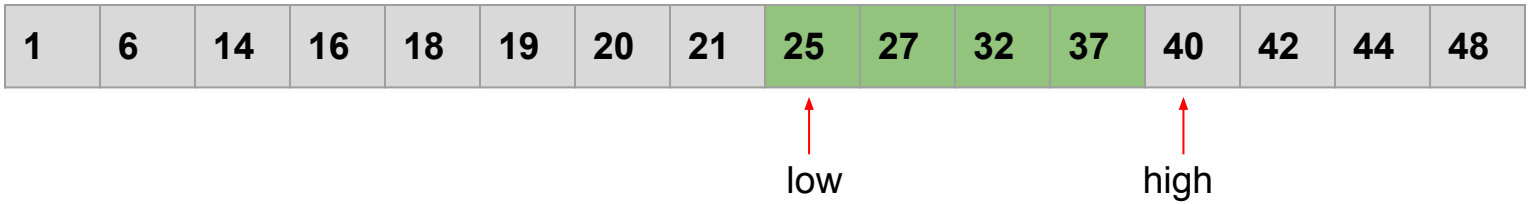
Binary search

Binary search

Binary search

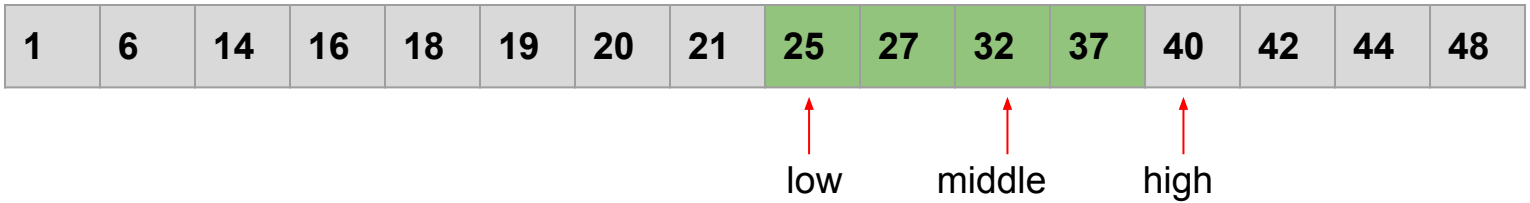
Binary search

```
int binarysearch(int key, const int array[], int size) {
  int low = 0;
  int high = size;

  // binary search algorithm


}
```



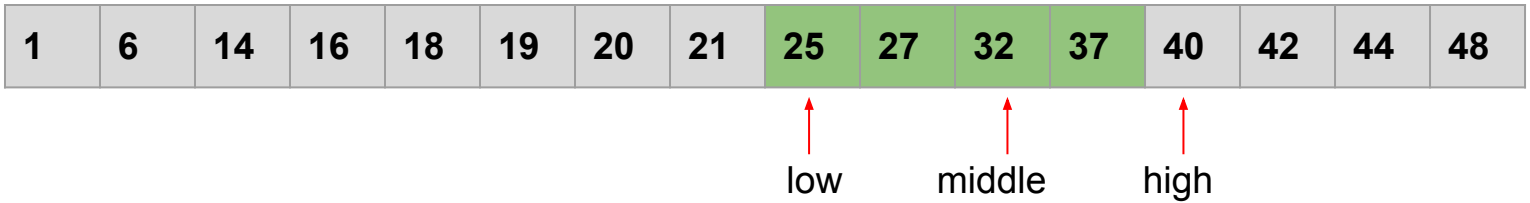
Binary search

```
int binarysearch(int key, const int array[], int size) {
  int low = 0;
  int high = size;

  while (???) {

    int middle = (high + low)/2;

    if (key < array[middle]) {

    }
    else {

    }

  }

}
```



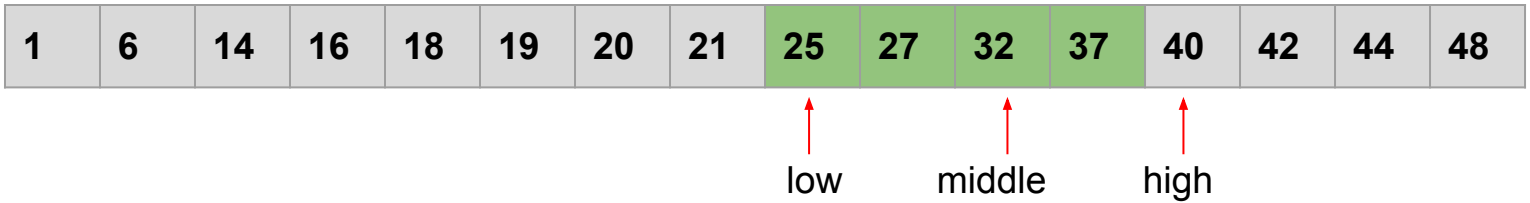
Binary search

```
int binarysearch(int key, const int array[], int size) {
  int low = 0;
  int high = size;

  while (???) {

    int middle = (high + low)/2;

    if (key < array[middle]) {
      high = middle;
    }
    else {
      low = middle;
    }

  }

}
```



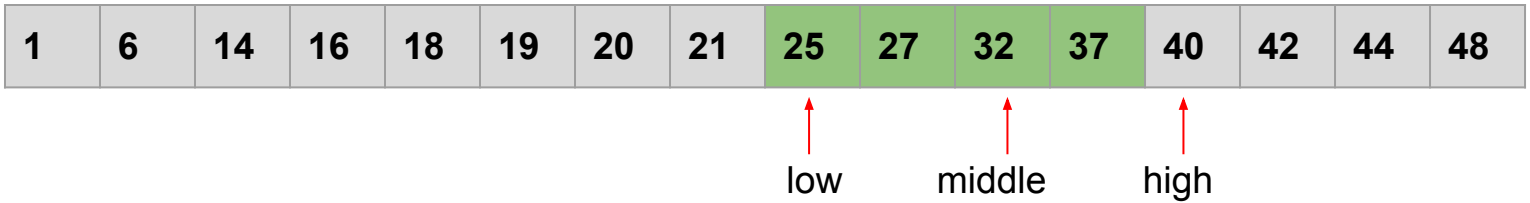
Binary search

```
int binarysearch(int key, const int array[], int size) {
  int low = 0;
  int high = size;

  while (high > low+1) {

    int middle = (high + low)/2;

    if (key < array[middle]) {
      high = middle;
    }
    else {
      low = middle;
    }

  }

}
```



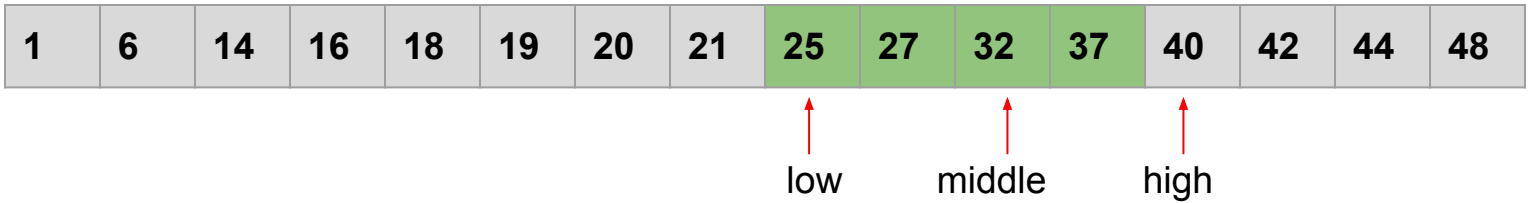
Binary search

```
int binarysearch(int key, const int array[], int size) {
  int low = 0;
  int high = size;

  while (high > low+1) {

    int middle = (high + low)/2;

    if (key < array[middle]) {
      high = middle;
    }
    else {
      low = middle;
    }

  }

  if (array[low] == key)
    return low;
  else
    return -1;

}
```



Binary search

```c
int binarysearch(int key, const int array[], int size) {
  int low = 0;
  int high = size;

  while (high > low+1) {

    int middle = (high + low)/2;

    if (key < array[middle]) {
      high = middle;
    }
    else {
      low = middle;
    }

  }

  return (array[low] == key) ? low : -1;

}
```



```c
int binarysearch(int key, const int array[], int size) {
  int low = 0;
  int high = size;

  while (high > low+1) {

    int middle = (high + low)/2;

    if (key < array[middle]) {
      high = middle;
    }
    else {
      low = middle;
    }

  }

  return (array[low] == key) ? low : -1;

}
```

- Assume size > 0
  - Can we ever have **high <= low**?
  - Why high = low + 1 out of the loop?
  - What if size == 1?
- What if size == 0?
- How many comparisons?
- Compare to linear search
  - size = 64
  - size = 1024
  - size ≈ 1000,000,000
- Sorting the array first?