2510 : Lecture 21 || March 10/17

- What is the prototype of free ?

both cases, we use free(fp) to deallocate memory

```
free (fp);
int *p = malloc (100 * sizeof (int));
double *p = malloc (50 * sizeof (double));
```

- void * pointers

↳ a void * pointer is compatible with any type of pointers to objects.

```
int n = 123;
int *p = &n;
void *q = &n;   /* ok... */
double *r = &n; /* compiler complains */
                  incorrect type
p = q;  ok
q = p;  ok
```

- A void * pointer is like a generic pointer
- this allows us to write functions that take any type of pointers

```
void free (void *);
void * malloc (size_t);
void * calloc (size_t, size_t);
void * realloc (void *, size_t);
```

- we cannot directly dereference a void * pointer; we either cast it to a pointer of the correct type and reference, or we assign it to a pointer of the correct type and dereference that other pointer.

```
int n = 123;
void *p = &n;
*p = 456; /* invalid */
```
CAST →
```
*(int *)p = 456; /* ok, changes n to 456 */
q = p; /* ok */
*q = 123; /* ok, changes n back to 123 */
```

- const
```
const int n = 1;
n = 2; /* invalid */
```

→ using const with pointers:

(int is const)

`const int *p`	p is a pointer to an int const
`int const *q`	q is a pointer to an const int
`int * const r`	r is a constant point to an int
`int *s const`	invalid

] same !

[read from right to left]

```
int n = 123;
int m = 456;
const int *p = &n;
*p = 234; /* invalid; can not use p to change the int */
p = &m; /* ok, makes p point to m */
int * const q = &n;
*q = 234; /* ok, changes n to 234 */
q = &m; /* invalid, q is const */
```

either this or that

Can also have const int * const r = &n
(both r and *r are constant)

... Lecture 21

- Array of Pointers

```
char *a[10]; /* array of 10 pointers to char */
char * b[] = {"hello", "world"};
printf("%s\n", b[0]); /* hello */
```

same

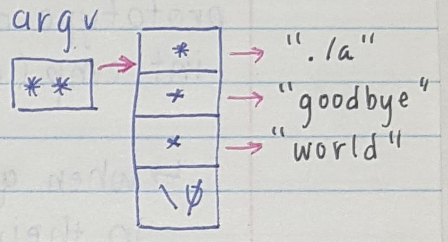
```
int main(int argc, char * argv[]){...}
```

```
$ ./a goodbye world
```

prompt

program name

```
argv[0] = "./a"
argv[1] = "goodbye"
argv[2] = "world"
```



$\underbrace{\text{argv[1]}}_{\text{char *}}\text{[2]}$ 3rd char of "goodbye"

- qsort

```
#include <stdlib.h>
```

-used to sort arrays

asn03

```
int a[100]; /* assume we have stored 100 ints in a */
qsort(a, 100, sizeof(a[0]), cmp);
```

comparison fCN: used to specify sorting order.

```
int cmp(const void *p, const void *q){
    const int *pp = p;
    const int *qq = q;
    return *pp - *qq; /* ascending order of ints */
}
```