strftime *Write Formatted Date and Time to String* <time.h>

```
size_t strftime(char * restrict s, size_t maxsize,
                const char * restrict format,
                const struct tm * restrict timeptr);
```

Stores characters in the array pointed to by `s` under control of the string pointed to by `format`. The format string may contain ordinary characters, which are copied unchanged, and conversion specifiers, which are replaced by values from the structure pointed to by `timeptr`. The `maxsize` parameter limits the number of characters (including the null character) that can be stored.

Returns Number of characters stored (not including the terminating null character). Returns zero if the number of characters to be stored (including the null character) exceeds `maxsize`. *26.3*

strlen *String Length* <string.h>

```
size_t strlen(const char *s);
```

Returns Length of the string pointed to by `s`, not including the null character. *13.5, 23.6*

strncat *Bounded String Concatenation* <string.h>

```
char *strncat(char * restrict s1,
              const char * restrict s2, size_t n);
```

Appends characters from the array pointed to by `s2` to the string pointed to by `s1`. Copying stops when a null character is encountered or `n` characters have been copied.

Returns `s1` (a pointer to the concatenated string). *13.5, 23.6*

strncmp *Bounded String Comparison* <string.h>

```
int strncmp(const char *s1, const char *s2, size_t n);
```

Returns A negative, zero, or positive integer, depending on whether the first `n` characters of the array pointed to by `s1` are less than, equal to, or greater than the first `n` characters of the array pointed to by `s2`. Comparison stops if a null character is encountered in either array. *23.6*

strncpy *Bounded String Copy* <string.h>

```
char *strncpy(char * restrict s1,
              const char * restrict s2, size_t n);
```

Copies the first `n` characters of the array pointed to by `s2` into the array pointed to by `s1`. If it encounters a null character in the array pointed to by `s2`, `strncpy` adds null characters to the array pointed to by `s1` until a total of `n` characters have been written.

Returns `s1` (a pointer to the destination). *13.5, 23.6*