

This sheet will not be graded (feel free to write on it).

C Function Definitions

```
size_t fread(void *ptr, size_t size, size_t nmemb, FILE *stream);
```

The function `fread()` reads `nmemb` items of data, each `size` bytes long, from the stream pointed to by `stream`, storing them at the location given by `ptr`. If `fread` receives an EOF before the total number of expected bytes has been read, it still stores the bytes read so far into `ptr`.

Note that `fread()` does not add a null byte after input.

```
int printf(const char *format, ...);
```

`printf()` produces output according to the format string `format`.

Conversion specifiers:

- `%c` Character.
- `%d` Signed integer.
- `%n` Writes the number of bytes printed so far, as a 4-byte integer, to the corresponding memory address.
- `%s` String
- `%u` Unsigned integer.
- `%x` Unsigned integer, in hexadecimal.

Each of the above conversion specifiers reads a 4-byte argument on the stack.

```
char *fgets(char *s, int size, FILE *stream);
```

`fgets()` reads in at most one less than `size` characters from `stream` and stores them into the buffer pointed to by `s`. Reading stops after an EOF or a newline. If a newline is read, it is stored into the buffer. A terminating null byte ('\0') is stored after the last character in the buffer.

```
void *memcpy(void *dest, const void * src, size_t n);
```

The `memcpy()` function copies `n` bytes from memory area `src` to memory area `dest`.

```
char *strncpy(char *dest, const char *src, size_t n);
```

The `strncpy()` function copies at most `n` characters from the string pointed to by `src` to the buffer pointed to by `dest`. If the source string is shorter than `n` characters, the remaining bytes in `dest` are padded with '\0'. If the source string is longer than or equal to `n` characters, no null byte is added.

```
char *gets(char *s);
```

`gets()` reads a line from `stdin` into the buffer pointed to by `s` until either a terminating newline or EOF, which it replaces with a null byte ('\0').

General Exam Assumptions

Unless otherwise specified, you can assume these facts on the entire exam:

- Memory Safety:
 - You are on a little-endian 32-bit x86 system.
 - There is no compiler padding or saved additional registers.
 - If stack canaries are enabled, they are four completely random bytes (no null byte).
 - You can write your answers in Python syntax (as seen in Project 1).
 - Unless otherwise specified, all other memory safety defenses are disabled.
 - Unless otherwise specified, each x86 instruction is 4-bytes long in machine code.
- Cryptography:
 - The attacker knows the algorithms being used (Shannon's maxim).
 - $\|$ denotes concatenation.
 - H refers to a secure cryptographic hash function.
 - g and p refer to a public generator element and large prime modulus, respectively.
 - IVs are randomly generated per encryption unless otherwise specified.
 - Enc refers to an IND-CPA secure encryption scheme, unless otherwise specified.
 - $\text{Generate}(n)$ generates n bits from a given PRNG.