Variadic functions

Variadic functions are functions (e.g. `printf`) which take a variable number of arguments.

The declaration of a variadic function uses an ellipsis as the last parameter, e.g. `int printf(const char* format, ...);`. See variadic arguments for additional detail on the syntax and automatic argument conversions.

Accessing the variadic arguments from the function body uses the following library facilities:

Macros

Defined in header <stdarg.h>

va_start	enables access to variadic function arguments (function macro)
va_arg	accesses the next variadic function argument (function macro)
va_copy (C99)	makes a copy of the variadic function arguments (function macro)
va_end	ends traversal of the variadic function arguments (function macro)

Type

va_list	holds the information needed by va_start, va_arg, va_end, and va_copy (typedef)

Example

Print values of different types.

Run this code

```
#include <stdio.h>
#include <stdarg.h>

void simple_printf(const char* fmt, ...)
{
    va_list args;
    va_start(args, fmt);

    while (*fmt != '\0') {
        if (*fmt == 'd') {
            int i = va_arg(args, int);
            printf("%d\n", i);
        } else if (*fmt == 'c') {
            // A 'char' variable will be promoted to 'int'
            // A character literal in C is already 'int' by itself
            int c = va_arg(args, int);
            printf("%c\n", c);
        } else if (*fmt == 'f') {
            double d = va_arg(args, double);
            printf("%f\n", d);
        }
        ++fmt;
    }

    va_end(args);
}

int main(void)
{
    simple_printf("dcff", 3, 'a', 1.999, 42.5);
}
```

Output:

```
3
a
1.999000
42.50000
```

References

- C11 standard (ISO/IEC 9899:2011):
  - 7.16 Variable arguments <stdarg.h> (p: 269-272)
- C99 standard (ISO/IEC 9899:1999):
  - 7.15 Variable arguments <stdarg.h> (p: 249-252)
- C89/C90 standard (ISO/IEC 9899:1990):
  - 4.8 VARIABLE ARGUMENTS <stdarg.h>

See also

C++ documentation for **`Variadic functions`**