JANUARY 22, 2020 / #C

Format Specifiers in C

Format specifiers define the type of data to be printed on standard output. You need to use format specifiers whether you're printing formatted output with `printf()` or accepting input with `scanf()`.

Some of the % specifiers that you can use in ANSI C are as follows:

SPECIFIER	USED FOR
%c	a single character
%s	a string
%hi	short (signed)
%hu	short (unsigned)
%Lf	long double
%n	prints nothing
%d	a decimal integer (assumes base 10)
%i	a decimal integer (detects the base automatically)
%o	an octal (base 8) integer

%p	an address (or pointer)
%f	a floating point number for floats
%u	int unsigned decimal
%e	a floating point number in scientific notation
%E	a floating point number in scientific notation
%%	the % symbol

Examples:

`%c` single character format specifier:

```
#include <stdio.h>

int main() {
  char first_ch = 'f';
  printf("%c\n", first_ch);
  return 0;
}
```

Output:

```
f
```

```
#include <stdio.h>

int main() {
  char str[] = "freeCodeCamp";
  printf("%s\n", str);
  return 0;
}
```

Output:

```
freeCodeCamp
```

Character input with the `%c` format specifier:

```
#include <stdio.h>

int main() {
  char user_ch;
  scanf("%c", &user_ch); // user inputs Y
  printf("%c\n", user_ch);
  return 0;
}
```

Output:

```
Y
```

```
#include <stdio.h>

int main() {
  char user_str[20];
  scanf("%s", user_str); // user inputs fCC
  printf("%s\n", user_str);
  return 0;
}
```

Output:

```
fCC
```

`%d` and `%i` decimal integer format specifiers:

```
#include <stdio.h>

int main() {
  int found = 2015, curr = 2020;
  printf("%d\n", found);
  printf("%i\n", curr);
  return 0;
}
```

Output:

`%f` and `%e` floating point number format specifiers:

```
#include <stdio.h>

int main() {
  float num = 19.99;
  printf("%f\n", num);
  printf("%e\n", num);
  return 0;
}
```

Output:

```
19.990000
1.999000e+01
```

`%o` octal integer format specifier:

```
#include <stdio.h>

int main() {
  int num = 31;
  printf("%o\n", num);
  return 0;
}
```

```
37
```

`%x` hexadecimal integer format specifier:

```
#include <stdio.h>

int main() {
  int c = 28;
  printf("%x\n", c);
  return 0;
}
```

Output:

```
1c
```