

Cartilla de código piola

UTN FRSF - El Rejunte

2016

Índice

1. C/C++1

1.1. Typedefs1

1.2. Macros1

1.3. I/O1

1.3.1. scanf Format Strings1

1.3.2. printf Format Strings2

1.3.3. Fast C++ Input3

2. Constantes y Tablas3

2.1. Constantes3

1. C/C++

1.1. Typedefs

```
1 typedef long long ll;
2 typedef long double ld;
3 typedef pair<int,int> pii;
4 typedef vector<pii> vpii;
5 typedef vector<int> vi;
6 typedef vector<vi> vvi;
```

1.2. Macros

```
1 //buffers
2 #define MEMZ(mem) memset(mem, 0, sizeof mem) // memset => <cstring / string.h>
3 #define MEMX(mem,x) memset(mem, (x), sizeof mem)
4
5 //loops
6 #define forall(i,a,b) for(int (i)=(a);(i)<(b);++(i))
7 #define foreach(i,v) for(typeof((v).begin()) (i) = (v).begin(); (i) != (v).end(); ++(i))
8
9 //map/pairs
10 #define mp make_pair
11 #define fi first
12 #define se second
13
14 //vector
15 #define pb push_back
16
17 //queries
18 #define in(a,v) ((v).find((a))!=(v).end()) // <algorithm>
```

1.3. I/O

1.3.1. scanf Format Strings

%[*][width][length]specifier

spec	Tipo	Descripción
i	int	Dígitos dec. [0-9], oct. (0) [0-7], hexa (0x 0X) [0-9a-fA-F]. Con signo.
d, u	int, unsigned	Dígitos dec. [+−0-9].
o	unsigned	Dígitos oct. [+−0-7].
x	unsigned	Dígitos hex. [+−0-9a-fA-F]. Prefijo 0x, 0X opcional.
f, e, g	float	Dígitos dec. c/punto flotante [+−.0-9]. Prefijo 0x, 0X y sufijo e, E opcionales.

Continuación		
spec	Tipo	Descripción
c, [width]c	char, char*	Siguiente carácter. Lee width chars y los almacena contiguamente. No agrega \0.
s	char*	Secuencia de chars hasta primer espacio. Agrega \0.
p	void*	Secuencia de chars que representa un puntero.
[chars]	Scanset, char*	Caracteres especificados entre corchetes.] debe ser primero en la lista, - primero o último. Agrega \0
[^chars]	!Scanset, char*	Caracteres no especificados entre corchetes.
n	int	No consume entrada. Almacena el número de chars leídos hasta el momento.
%		%% consume un %

sub-specifier	Descripción
*	Indica que se leerá el dato pero se ignorará. No necesita argumento.
width	Cantidad máxima de caracteres a leer.
length	Uno de hh, h, l, ll, j, z, t, L. Ver tabla siguiente.

length (none)	d i	u o x
hh	int*	unsigned int*
h	signed char*	unsigned char*
h	short int*	unsigned short int*
l	long int*	unsigned long int*
ll	long long int*	unsigned long long int*
j	intmax_t*	uintmax_t*
z	size_t*	size_t*
t	ptrdiff_t*	ptrdiff_t*
L		

length (none)	f e g a	c s [] [^]	p	n
hh	float*	char*	void**	int*
h				signed char*
h				short int*
l	double*	wchar_t*		long int*
ll				long long int*
j				intmax_t*
z				size_t*
t				ptrdiff_t*
L	long double*			

1.3.2. printf Format Strings

%[flags][width][.precision][length]specifier

specifier	Descripción	Ejemplo
d or i	Entero decimal con signo	392
u	Entero decimal sin signo	7235
o	Entero octal sin signo	610

Continuación		
specifier	Descripción	Ejemplo
x	Entero hexadecimal sin signo	7fa
X	Entero hexadecimal sin signo (mayúsculas)	7FA
f	Decimal punto flotante (minúsculas)	392.65
F	Decimal punto flotante (mayúsculas)	392.65
e	Notación científica (mantisa/exponente), (minúsculas)	3.9265e+2
E	Notación científica (mantisa/exponente), (mayúsculas)	3.9265E+2
g	Utilizar la representación más corta: %e ó %f	392.65
G	Utilizar la representación más corta: %E ó %F	392.65
a	Hexadecimal punto flotante (minúsculas)	-0xc.90fep-2
A	Hexadecimal punto flotante (mayúsculas)	-0XC.90FEP-2
c	Caracter	a
s	String de caracteres	sample
p	Dirección de puntero	b8000000
n	No imprime nada. El argumento debe ser int*, almacena el número de caracteres imprimidos hasta el momento.	
%	Un % seguido de otro % imprime un solo %	%

flag	Descripción
-	Justificación a la izquierda dentro del campo width (ver width sub-specifier).
+	Forza a preceder el resultado de texttt+ o texttt-.
(espacio)	Si no se va a escribir un signo, se inserta un espacio antes del valor.
#	Usado con o, x, X specifiers el valor es precedido por 0, 0x, 0X respectivamente para valores distintos de 0.
0	Rellena el número con texttt0 a la izquierda en lugar de espacios cuando se especifica width.

width	Descripción
(número)	Número mínimo de caracteres a imprimir. Si el valor es menor que número, el resultado es rellando con espacios. Si el valor es mayor, no es truncado.
*	No se especifica width, pero se agrega un argumento entero precediendo al argumento a ser formateado. Ej. printf(“---%d---\n”, 3, 2); ⇒ “---- 5----”.

precision	Descripción
	Para d, i, o, u, x, X: número mínimo de dígitos a imprimir. Si el valor es más chico que número se rellena con 0.
.	Para a, A, e, E, f, F: número de dígitos a imprimir después de la coma (default 6).
	Para g, G: Número máximo de cifras significativas a imprimir.
	Para s: Número máximo de caracteres a imprimir. Trunca.

Continuación	
precision	Descripción
. *	No se especifica precision pero se agrega un argumento entero precediendo al argumento a ser formateado.

length	d i	u o x X
(none)	int	unsigned int
hh	signed char	unsigned char
h	short int	unsigned short int
l	long int	unsigned long int
ll	long long int	unsigned long long int
j	intmax_t	uintmax_t
z	size_t	size_t
t	ptrdiff_t	ptrdiff_t
L		

length	f F e E g G a A	c	s	p	n
(none)	double	int	char*	void*	int*
hh					signed char*
h					short int*
l		wint_t	wchar_t*		long int*
ll					long long int*
j					intmax_t*
z					size_t*
t					ptrdiff_t*
L	long double				

1.3.3. Fast C++ Input

```
1 ios_base::sync_with_stdio(false); cin.tie(NULL);
```

2. Constantes y Tablas

2.1. Constantes

```
1 #define INF 10000000000 // 1 billon, entra en int
2 #define EPS 1e-12
3 #define PI 3.1415926535897932384626
```