C History

- Designed by Dennis Ritchie and Ken Thompson at Bell Labs in 1972.
- It was based in B, a squeezed down minimal subset of BCPL implemented on an 8K PDP-7 by Ken Thompson.
- In 1970 the Unix project acquired a 24K PDP-11 and B was expanded to include structures and a more operators. This language became known as C.
- It was first used to write the kernel of Unix and has been closely associated with Unix ever since.
- ANSI Standard (1989)
- ISO Standard (1990)

C Example

```c
#include <stdio.h>

const int maxsize=9;

int convert(char ch) { return ch-'0'; }

int sum(int v[], int n) {
  int res=0;
  int j;
  for(j=0; j<n; j++) res+=v[j];
  return res;
}

int main() {
  int a[maxsize];
  int j, k;
  while( (k=convert(getchar())) != 0) {
    for(j=0; j<k; j++) a[j]=convert(getchar());
    for(j=0; j<k; j++) printf("%d ", a[j]);
    printf("; SUM=%d\n", sum(a, k));
    while(getchar() != '\n');
  }
}
```

Compiling and Running

```
clove$ cc -o example example.c
clove$ example
41234
1 2 3 4 ; SUM 10
512345
1 2 3 4 5 ; SUM 15
0
clove$
```

C Primitive Types

Numeric types:

```
unsigned char
unsigned short
unsigned int
unsigned long
char
short
int
long

float
double
long double
```

C Primitive Types

Numeric constants:

```
'A'             -- int
123U            -- unsigned int
1234567890UL    -- unsigned long

123             -- int
0xFF0037        -- int
1234567890L     -- long



123.456F        -- float
123.456         -- double
123.456e-5      -- double
123.456E12      -- double
123.456L        -- long double
```

Monadic Expression Operators

```
e(e,e,...,e)     -- function call
e[e,e,...,e]     -- subscripted expression
e->name          -- structure selector
e.name           -- field selector

!e               -- boolean not
~e               -- bitwise not
++e    e++       -- pre/post increment
--e    e--       -- pre/post decrement
+e               -- monadic plus
-e               -- monadic minus
(type)e          -- cast (or type conversion)
*e               -- indirection
&e               -- address of
sizeof e         -- size of the given value
```

Infixed Expression Operators

```
e * e            -- multiplication
e / e            -- division
e % e            -- remainder
e + e            -- plus
e - e            -- minus

e << e           -- left shift
e >> e           -- right shift
```

Infixed Expression Operators

```
e<e   e>e   e<=e   e>=e     -- relations
e==e   e!=e                 -- relations

e & e          -- bitwize and
e ^ e          -- bitwize xor
e | e          -- bitwize or
e && e         -- boolean and
e || e         -- boolean or
e ? e : e      -- conditional expression
e=e  e+=e ...  -- assignments
e,e,...,e      -- expression list
```

Statements

```
{ d d ... d s s ... s }  -- a block
e;
;
if (e) s
if (e) s else s
if (e) s else s
while (e) s
do s while(e);
for(e;e;e)s  -- eg for(i=0;i<k;i++)v[i]=0;
switch(e) { case k: s
            ...
            case k: s
            default: s
          }
break;
continue;
return e;
goto name;
name: s
```

Declarations

```
int i, j=1, k;
float x;
long a, b;

char v[10];
int w[4] = { 0, 1, 2, 3};

int *p, **q;
int * tab[256];     -- vector of pointers

int f(int,int);     -- function declaration
int (*f)(int,int); -- pointer to a function
```

Casts

declaration	cast
`unsigned int i;`	`(unsigned int)`
`float x;`	`(float)`
`char v[10];`	`(char [10])`
`int w[4];`	`(int [4])`
`int *p;`	`(int *)`
`int **q;`	`(int **)`
`int * tab[256];`	`(int *[256])`
`int f(int,int);`	`(int(int,int))`
`int (*f)(int,int);`	`(int (*)(int,int))`
`void (*g)(void);`	`(void(*)(void))`

Functions

Declarations, eg:

```
/* a declaration  */
int mymax(int, int);
```

Definitions, eg:

```
/* a definition */
int mymax3(int a, int b, int c) {
  return mymax(a, mymax(b, c));
}

/* another definition */
int mymax(int a, int b) {
  return a>b ? a : b;
}
```