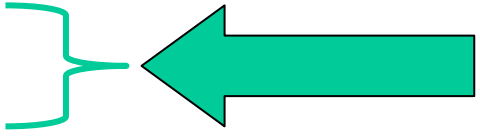
- La rappresentazione dell'Informazione
- Struttura dei programi C
  - Tipi Semplici del C
  - Costanti Variabili, Operatori
  - Espressioni
  - Operatori aritmetici, logici, bitwise



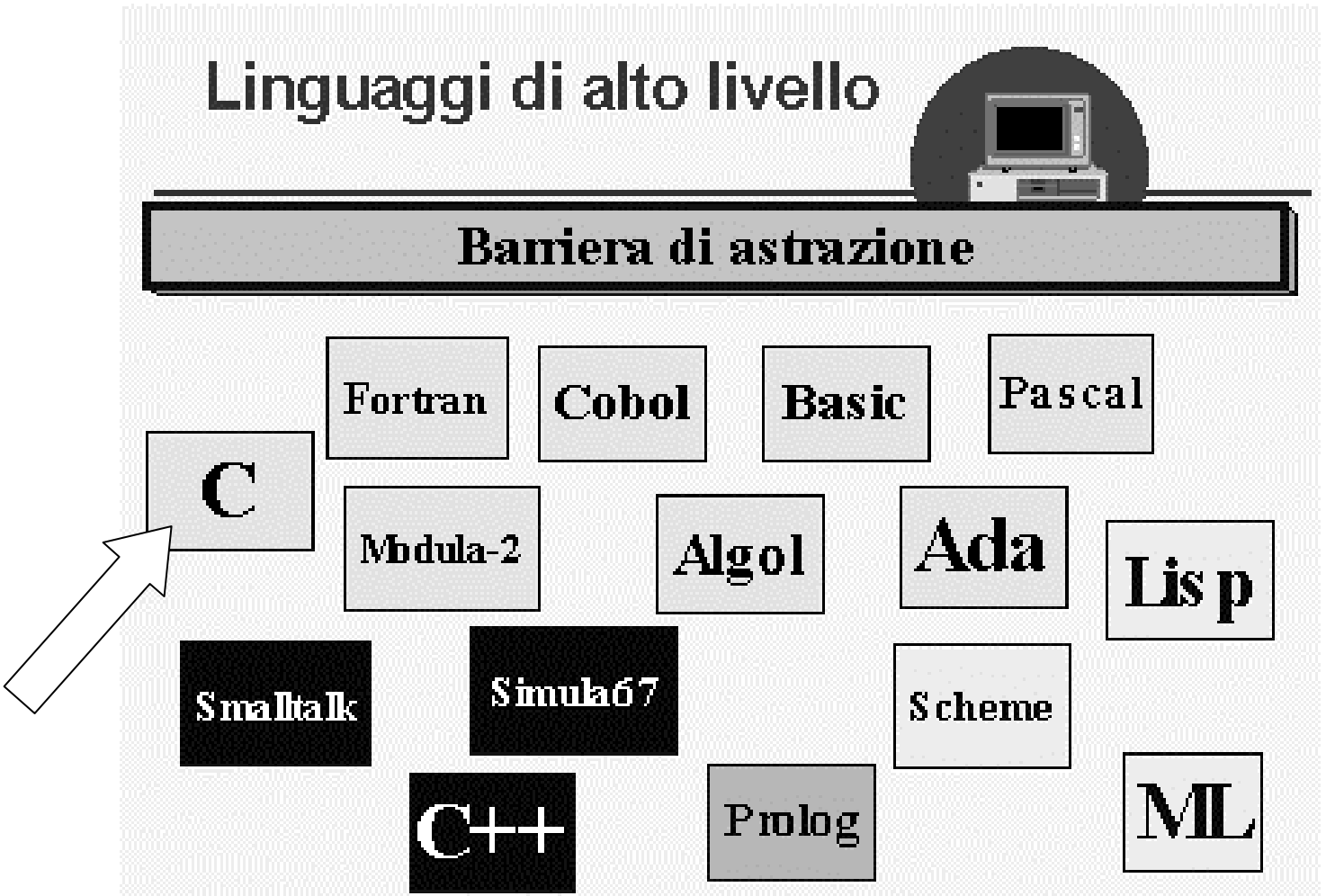
Passi fondamentali del C

- Definito nel 1972 (AT&T Bell Labs) per sostituire l’assembler nella programmazione di sistemi operativi: in pratica, nato per creare **UNIX**

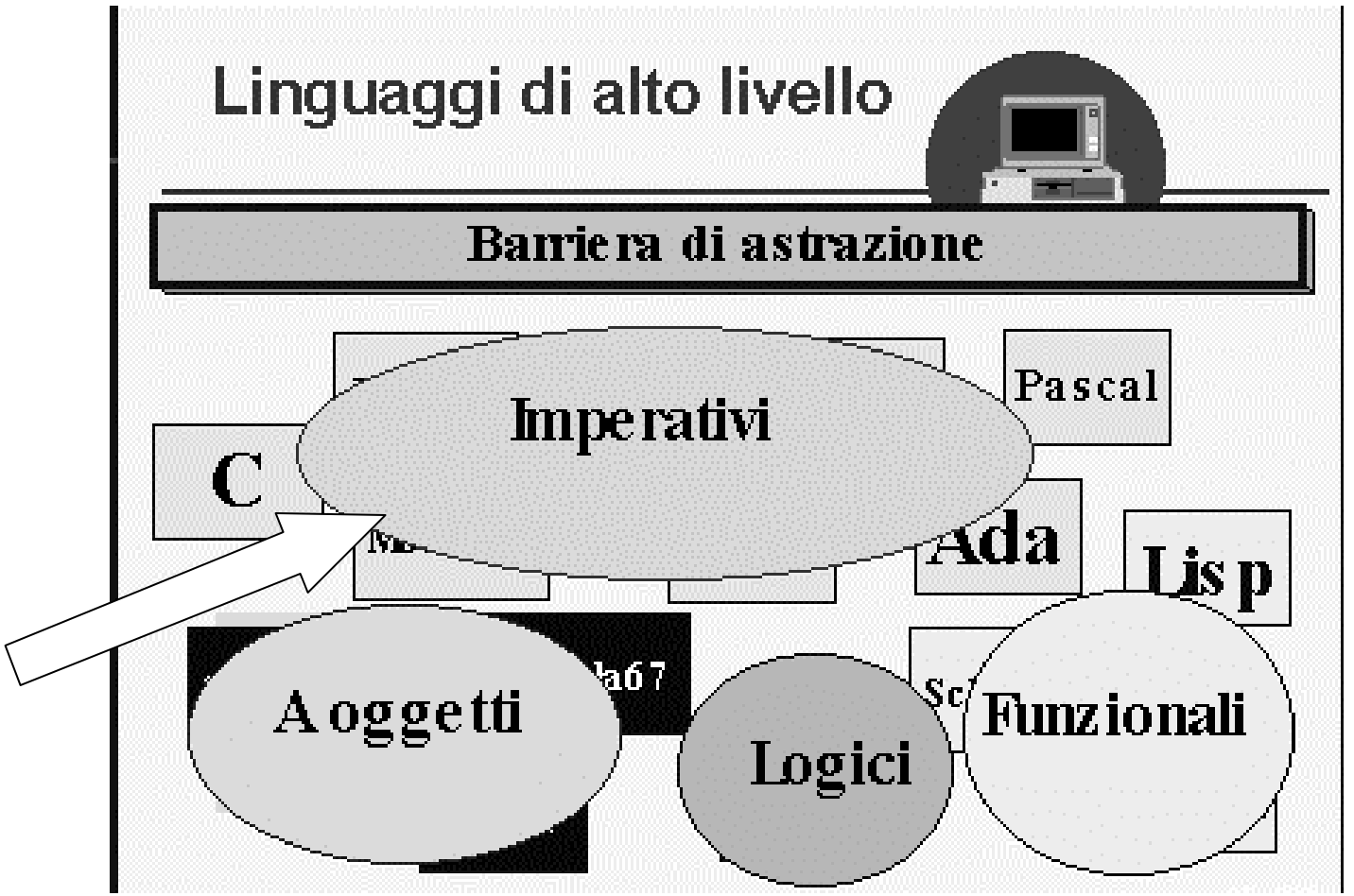
- Prima definizione precisa: Kernigham & Ritchie (1978)

- Prima definizione ufficiale: **ANSI C** (1983)

C: linguaggio di alto livello



C: linguaggio imperativo



Caratteristiche del C

- Linguaggio *sequenziale*, ***imperativo***, ***strutturato a blocchi***
- Usabile anche come linguaggio di sistema → adatto a software di base, sistemi operativi, compilatori, ecc.
- Portabile, efficiente, sintetico (ma a volte poco leggibile...)

- Basato su pochi concetti elementari:
  - **espressione**
  - **dichiarazione / definizione**
  - **istruzione / blocco**
  - **funzione**
- tuttavia, viene arricchito da un vasto insieme di librerie di funzioni (per operazioni matematiche, di input/output, su stringhe, ecc.)

Caratteristiche del C (*cont*.)

SET DI CARATTERI

Dipendente dalla implementazione (in genere ASCII più estensioni)

IDENTIFICATORI

<Identificatore> ::= <Lettera> { <Lettera> | <Cifra> }

- **Lettera** include tutte le lettere, maiuscole e minuscole, e l'underscore **"_"** (utilizzabile all'inizio solo per *oggetti di sistema*)

- Maiuscole e minuscole sono *diverse* (linguaggio C è *case-sensitive*)

PAROLE RISERVATE

- Esempio: **int, float, if, for, do, ...**

  { } delimitatore di blocco

COMMENTI

/* commento, anche su più righe */ (non possono essere innestati)

ELEMENTI DEL LINGUAGGIO C

parole chiave *(riservate)* ⇒ proprie del linguaggio
- sono le istruzioni oppure hanno un significato particolare (tipi)

identificatori ⇒ costituiti da sequenze di lettere ...
- rappresentano nomi di variabili, costanti, (tipi nuovi), funzioni, procedure
- definiti dall'utente oppure «di sistema» (di libreria)

operatori (unari o binari)
- di assegnamento =
- aritmetici + - * /
- relazionali (confronto) > < <= == ...
- logici ! (not) && ||
- di chiamata di funzione/procedura ( )
- di dereferenziazione &
- dipendenti dal costruttore di tipo * [ ] ....
- altri

separatori (delimitatori)
- di identificatori di variabili e costanti ,
- di istruzioni ;
- commento /* */
- di blocco di istruzioni { }
- in espressioni ( )

direttive al preprocessore C
- # include (parola chiave) .....

valori costanti (cifre o caratteri)

THE ‘C’ ALPHABET

The characters in ‘C’ are divided into 3 groups

- Digits: 0 - 9
- Letters: A - Z, a - z, $, -
- Special Characters:

b Blank	/ Slash	‘ Apostrophe
+ Plus	! Exclamation	* Asterisk
? Question Mark	^ Circumflex	(Let Parent
: Colon	| Stroke	) Right Par
< Less than	, Comma	[Left Bracket
= Equal	% Percent	] Right Bracket
> Greater than	$ Dollar sign	{ Left Brace
~ Tilde	. Period	} Right Brace
& Ampersand	_ Underscore	# Pound sign
- Hyphen	“ Quotation mark	; Semicolon
\ Backslash		

Reserved Words

auto	extern	sizeof
break	for	static
case	float	struct
char	goto	switch
const	if	typedef
continue	int	union
default	long	unsigned
do	register	void
double	return	volatile
else	short	while
enum	signed	

no
ok

Reserved Words

auto	extern	sizeof
break	for	static
case	float	struct
char	goto	switch
const	if	typedef
continue	int	union
default	long	unsigned
do	register	void
double	return	volatile
else	short	while
enum	signed	

Sintassi C: Frasi

Summary of Statements

statement :

compound-statement
expression-statement
selection-statement
iteration-statement
jump-statement

jump-statement :

continue;
break;
return *expression* $_{opt}$ **;**

compound-statement :
{ *declaration-list* $_{opt}$ *statement-list* $_{opt}$ **}**

declaration-list :
declaration
declaration-list declaration

statement-list :
statement
statement-list statement

expression-statement :
expression $_{opt}$ **;**

iteration-statement :
while (*expression* **)** *statement*
do *statement* **while (** *expression* **);**
for (*expression* $_{opt}$ **;** *expression* $_{opt}$ **;** *expression* $_{opt}$ **)** *statement*

selection-statement :
if (*expression* **)** *statement*
if (*expression* **)** *statement* **else** *statement*
switch (*expression* **)** *statement*

labeled-statement :

case *constant-expression* **:** *statement*
default : *statement*

Sintassi C: Frasi

Summary of Statements

statement :

compound-statement
expression-statement
selection-statement
iteration-statement
jump-statement

jump-statement :

continue;
break;
return *expression* $_{opt}$ **;**

compound-statement :
{ *declaration-list* $_{opt}$ *statement-list* $_{opt}$ **}**

declaration-list :
declaration
declaration-list declaration

statement-list :
statement
statement-list statement

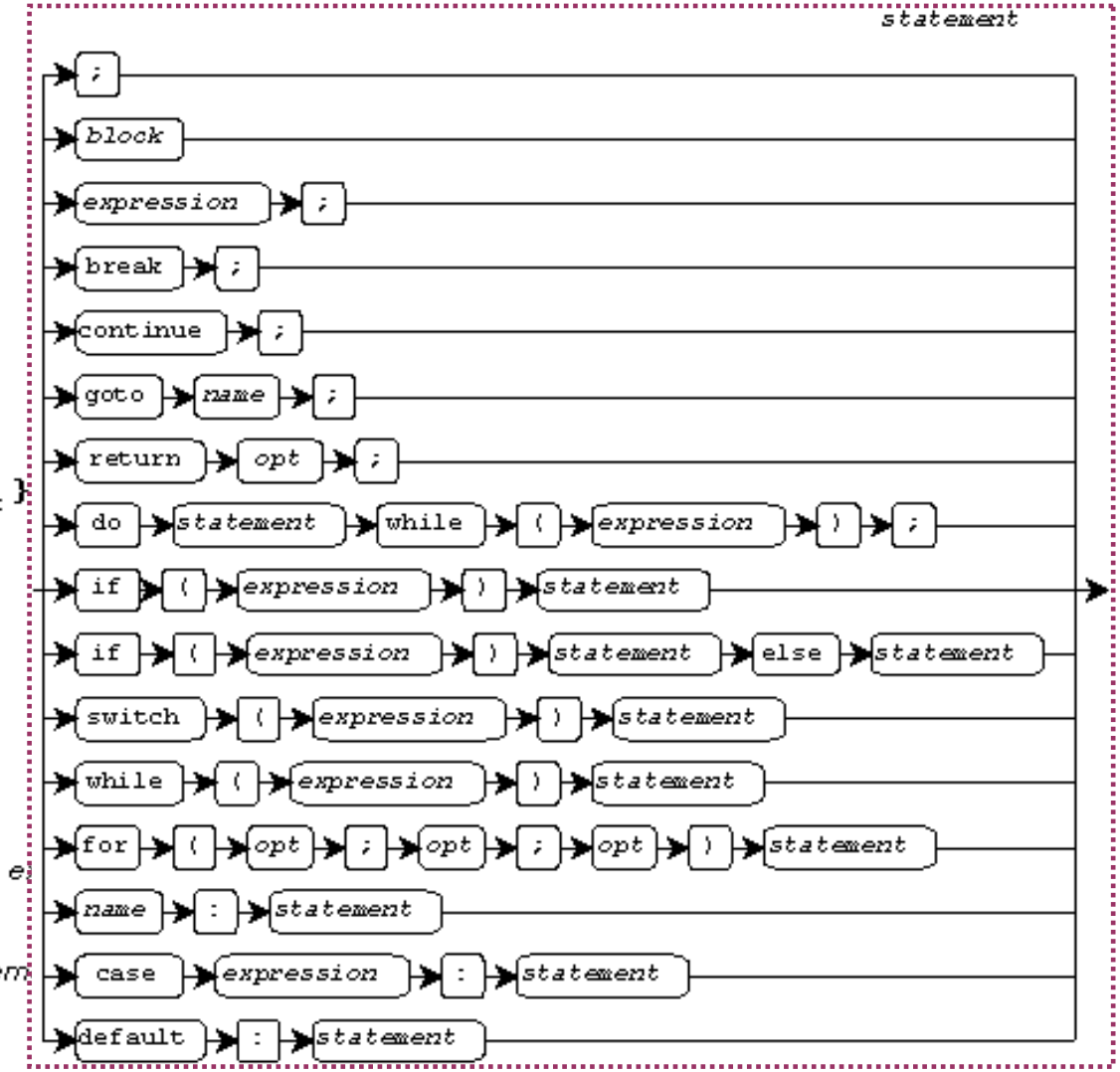
expression-statement :
expression $_{opt}$ **;**

iteration-statement :
while (*expression* **)** *statement*
do *statement* **while (** *expression* **);**
for (*expression* $_{opt}$ **;** *expression* $_{opt}$ **;** *e*

selection-statement :
if (*expression* **)** *statement*
if (*expression* **)** *statement* **else** *statem*
switch (*expression* **)** *statement*

labeled-statement :

case *constant-expression* **:** *statement*
default : *statement*



Struttura di un Programma C

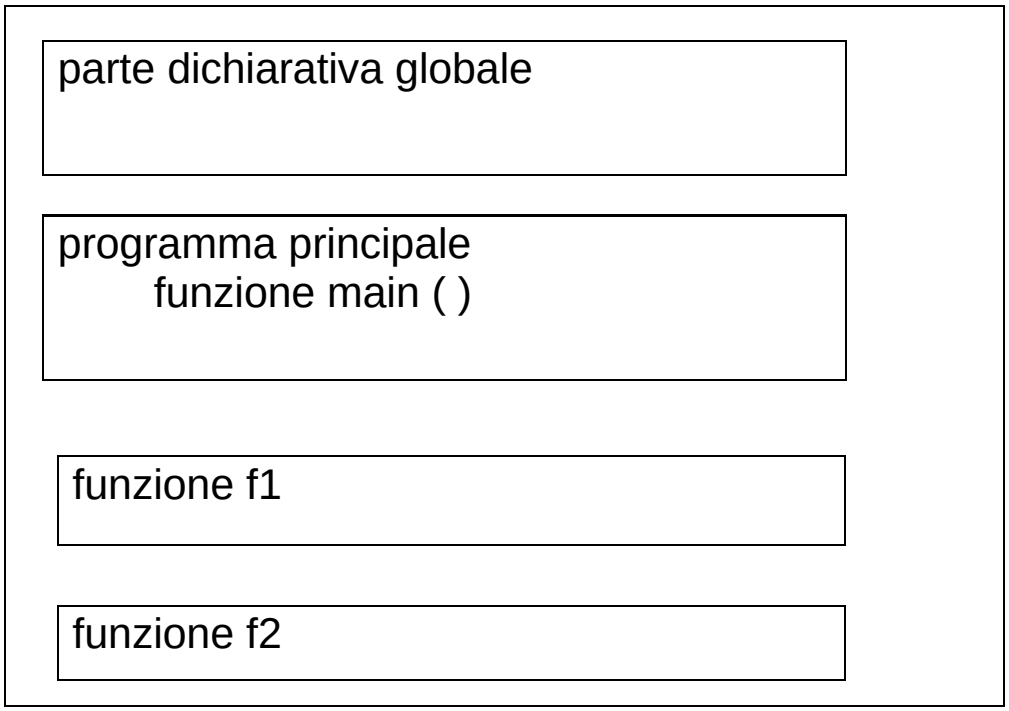
Un programma C ha in linea di principio la seguente forma:

- **Direttive per il preprocessore**
- **Definizione di tipi**
- **Prototipi di funzioni**, con dichiarazione dei tipi delle funzioni e delle variabili passate alle funzioni)
- **Dichiarazione delle Variabili Globali**
- **Dichiarazione Funzioni**, dove ogni dichiarazione di una funzione ha la forma:
  Tipo NomeFunzione(Parametri)
  {
  Dichiarazione Variabili Locali
  Istruzioni C
  }

```
#include <stdio.h>

typedef struct point {
            int   x;    int   y;
            };
int       f1(void);
void    f2(int i, double g);


int   sum;


int main(void)
{
      int j;
      double g=0.0;
      for(j=0;j<2;j++)
            f2(j,g);
      return(2);
}

void f2(int i, double g)
{
      sum = sum + g*i;
}
```

STRUTTURA DI UN PROGRAMMA C - 1

programma C



Stile di scrittura di un programma ⇒ **leggibilità**

- scelta degli identificatori
- indentazione: «struttura grafica» che rispecchia la struttura logica

- uso di commenti

STRUTTURA DI UN PROGRAMMA C - 2

Parte dichiarativa globale:

- servizi (funzioni) importate da altri moduli (file), cioè definite e codificate in altri file

- «oggetti» (tipi di dati, variabili, costanti simboliche, prototipi di funzioni) visibili (utilizzabili) da tutto il programma, cioè da main e dalle altre funzioni.

Programma principale:

```
main ()

{

  parte dichiarativa
  locale



  parte esecutiva

}
```

parola riservata (identificatore di funzione)
appare una e una sola volta nel programma
definisce l'inizio dell'esecuzione
è (formalmente) una funzione

definisce l'insieme di «oggetti» usati dal programma principale per l'esecuzione. sono oggetti visibili (locali) a main.

insieme di istruzioni che costituiscono il programma principale

STRUTTURA DI UN PROGRAMMA C - 3

Parte dichiarativa locale

1. dichiarazione di costanti
2. definizione di «nuovi» tipi definiti dall'utente (ridenominazione)
3. dichiarazione di variabili
4. prototipi di funzioni

«Regole» sintattiche sulle dichiarazioni sia locali che globali:

- ogni identificatore usato deve essere prima definito
- ogni variabile usata deve essere prima dichiarata

Parte esecutiva: istruzioni (per tipologia)

- istruzioni di assegnamento
- istruzioni composte
- costrutti di (modifica del flusso di) controllo (costrutti condizionali, costrutti ciclici)
- «istruzioni» di ingresso e uscita
- chiamate di sottoprogrammi (funzioni)

ESEMPIO 1 - Dichiarazioni e istruzioni

```
#include <stdio.h>


main ( )

{
 int potenza, base;
 int esponente, i;          /* devono essere >=0*/

 printf("Inserisci il valore della base:");
 scanf("%d",&base);
 printf("Inserisci il valore dell'esponente:");
 scanf("%d", &esponente);

 potenza=1;
 i=0;

 while (i<esponente)
   {
   potenza=potenza*base;
   i=i+1;
    }

 printf("Potenza=%d \n", potenza);
}
```

ESEMPIO 1 - Dichiarazioni e istruzioni

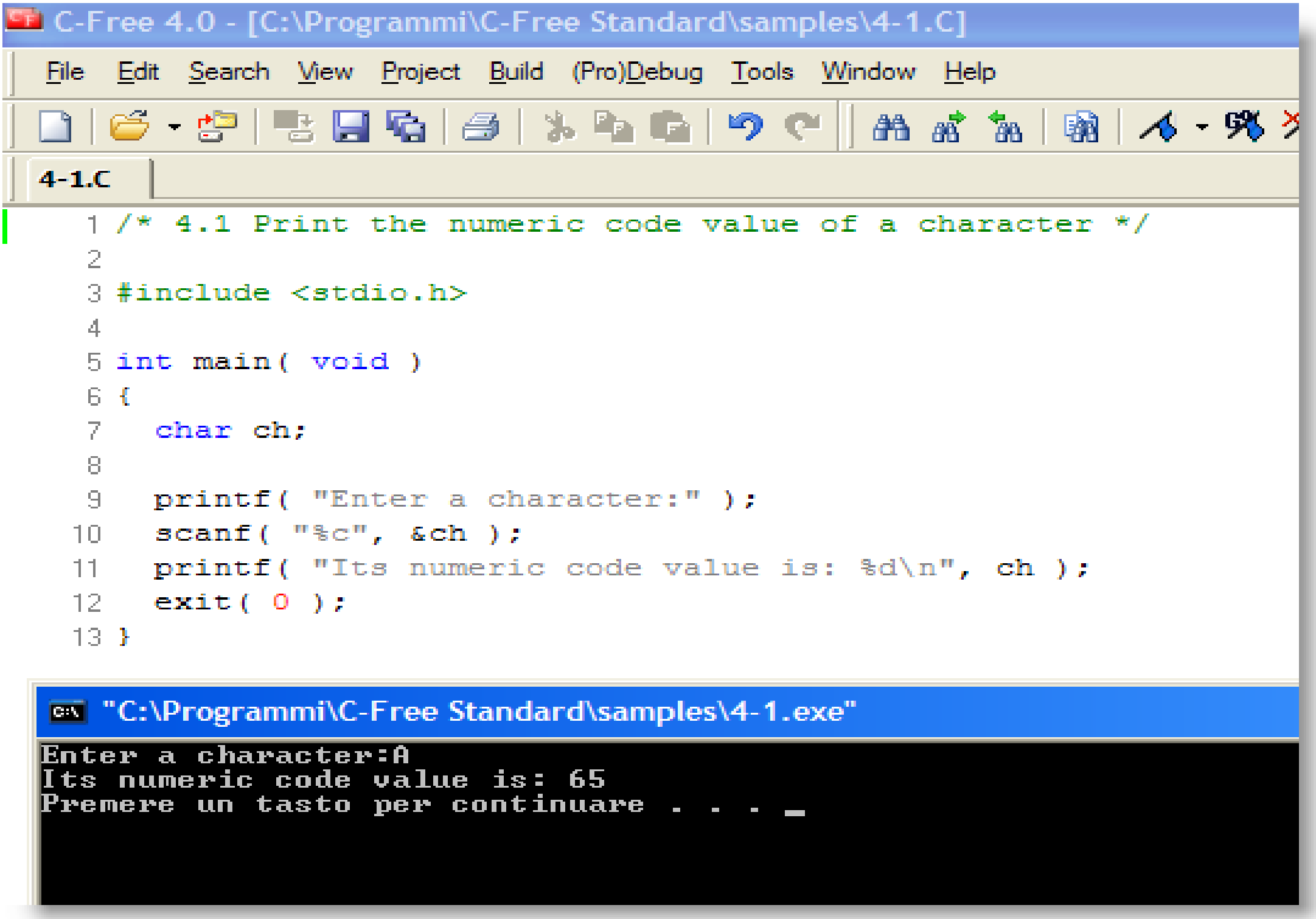
```
#include <stdio.h>


main ( )

{
 int potenza, base;
 int esponente, i;          /* devono essere >=0*/

 printf("Inserisci il valore della base:");
 scanf("%d",&base);
 printf("Inserisci il valore dell'esponente:");
 scanf("%d", &esponente);

 potenza=1;
 i=0;

 while (i<esponente)
   {
   potenza=potenza*base;
   i=i+1;
    }

 printf("Potenza=%d \n", potenza);
}
```

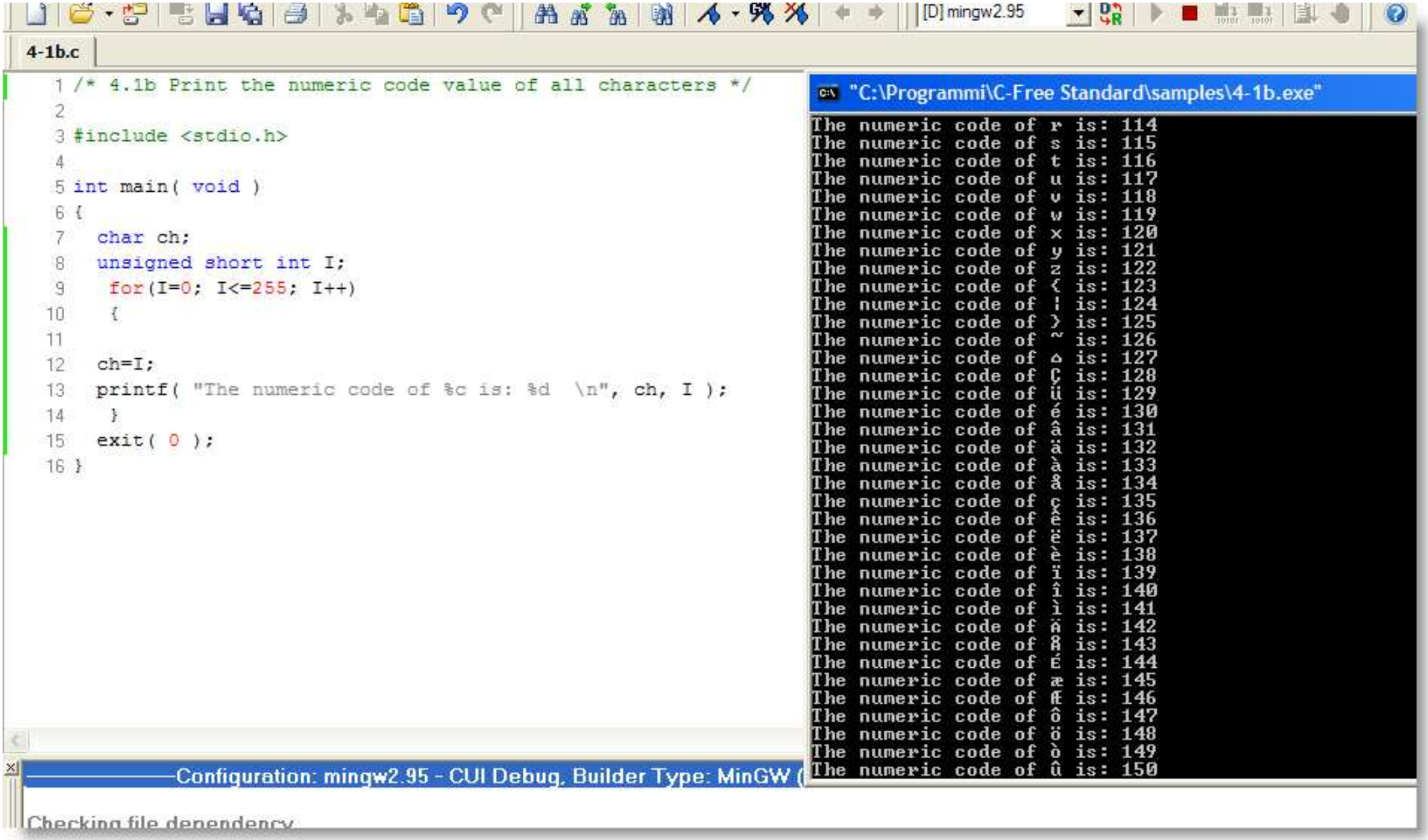
Template progr C

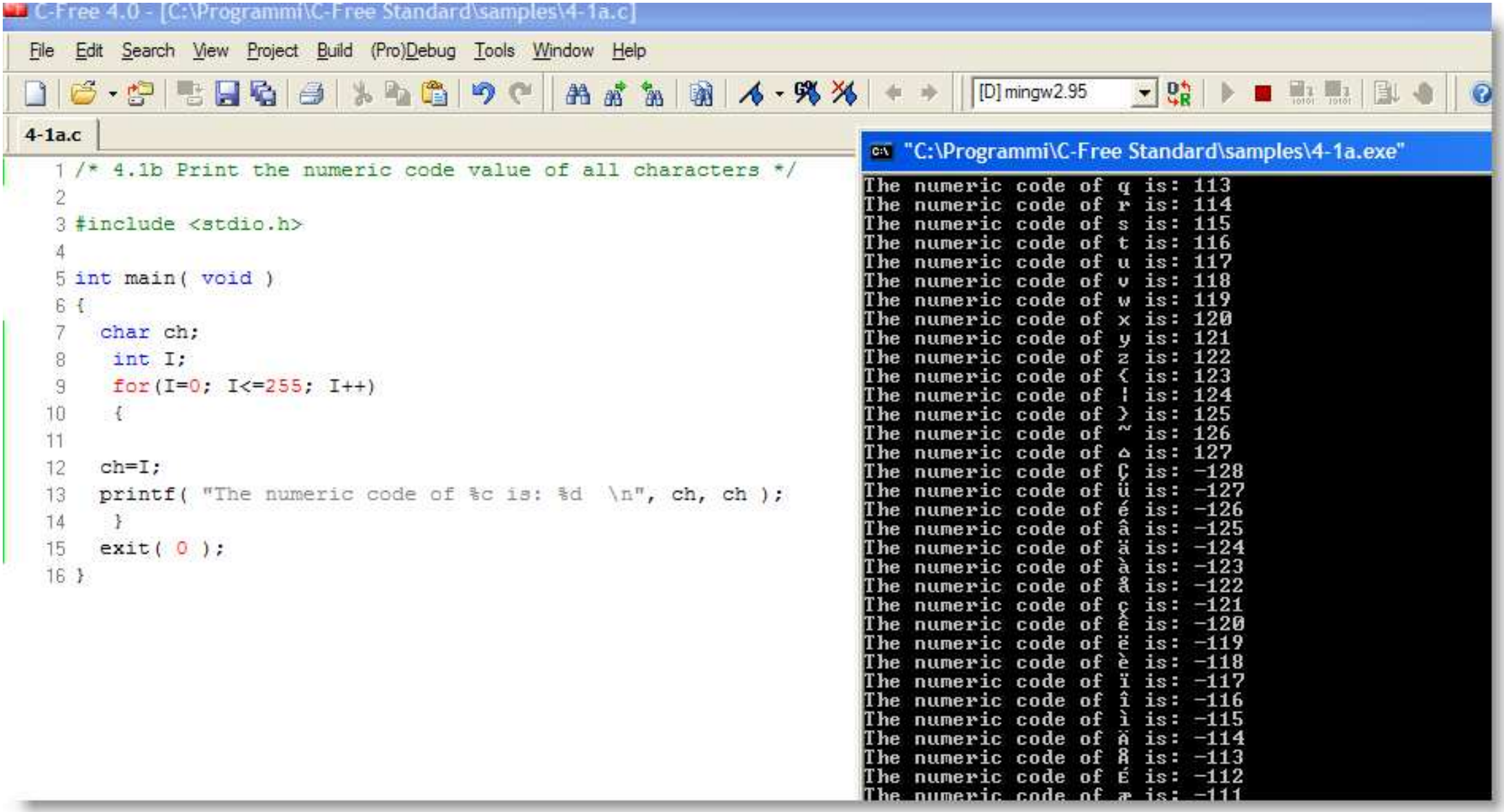
```
1 /**
2     Module Name:
3
4     Description:
5
6     Author:
7     Created:
8     Last Change:
9
10    Functions:
11
12 */
```

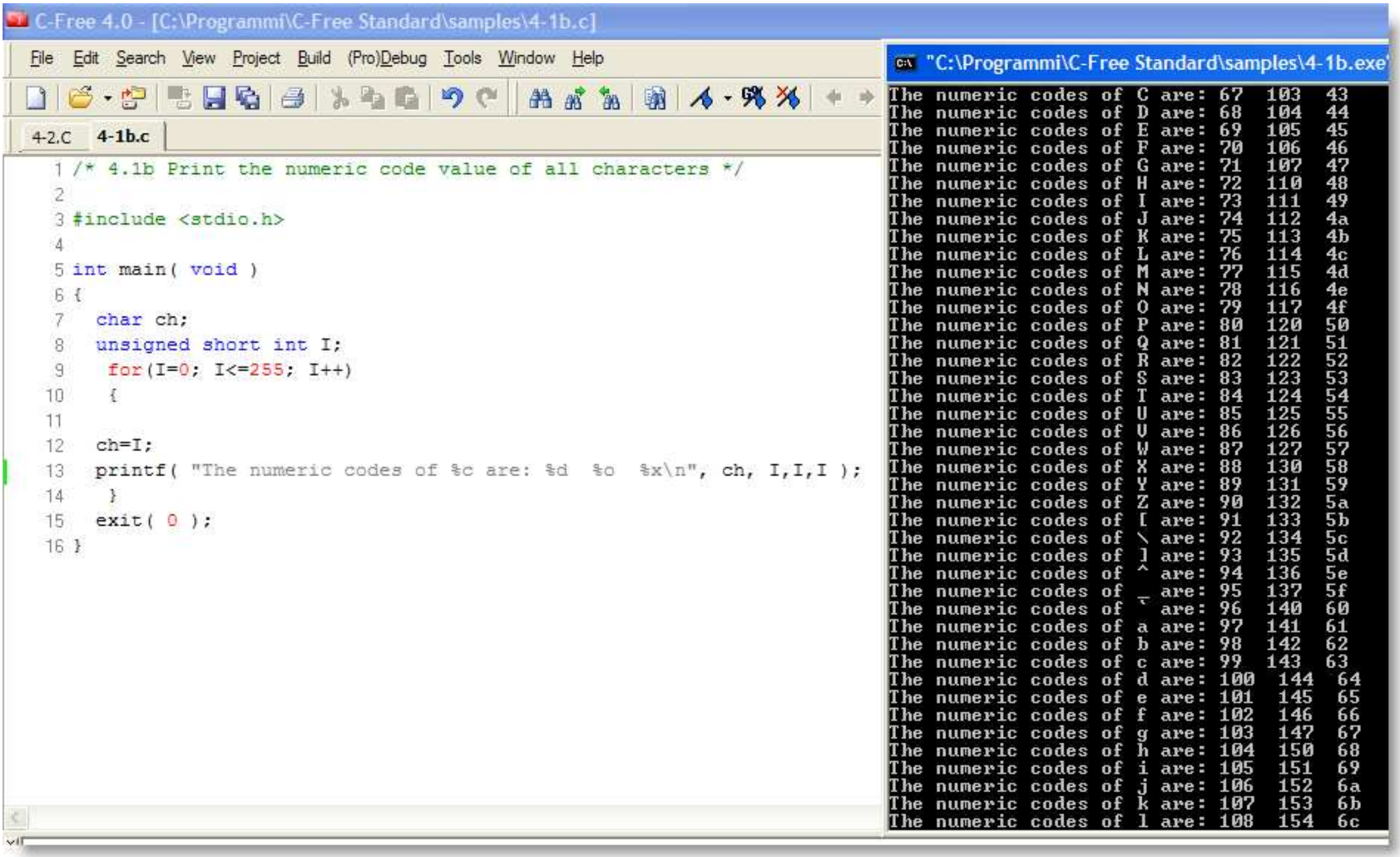
```
1 #include <stdio.h>
2 int main(int argc, char *argv[])
3 {
4
5     return 0;
6 }
```

programma


C-Free 4.0 - [C:\Programmi\C-Free Standard\samples\4-1.C]
File Edit Search View Project Build (Pro)Debug Tools Window Help
4-1.C
1 /* 4.1 Print the numeric code value of a character */
2
3 #include <stdio.h>
4
5 int main(void)
6 {
7 char ch;
8
9 printf("Enter a character:");
10 scanf("%c", &ch);
11 printf("Its numeric code value is: %d\n", ch);
12 exit(0);
13 }
"C:\Programmi\C-Free Standard\samples\4-1.exe"
Enter a character:A
Its numeric code value is: 65
Premere un tasto per continuare . . .


[D] mingw2.95
4-1b.c
1 /* 4.1b Print the numeric code value of all characters */
2
3 #include <stdio.h>
4
5 int main(void)
6 {
7 char ch;
8 unsigned short int I;
9 for(I=0; I<=255; I++)
10 {
11
12 ch=I;
13 printf("The numeric code of %c is: %d \n", ch, I);
14 }
15 exit(0);
16 }
"C:\Programmi\C-Free Standard\samples\4-1b.exe"
The numeric code of r is: 114
The numeric code of s is: 115
The numeric code of t is: 116
The numeric code of u is: 117
The numeric code of v is: 118
The numeric code of w is: 119
The numeric code of x is: 120
The numeric code of y is: 121
The numeric code of z is: 122
The numeric code of { is: 123
The numeric code of ! is: 124
The numeric code of } is: 125
The numeric code of ~ is: 126
The numeric code of ⌂ is: 127
The numeric code of Ç is: 128
The numeric code of ü is: 129
The numeric code of é is: 130
The numeric code of â is: 131
The numeric code of ä is: 132
The numeric code of à is: 133
The numeric code of å is: 134
The numeric code of ç is: 135
The numeric code of ê is: 136
The numeric code of ë is: 137
The numeric code of è is: 138
The numeric code of ï is: 139
The numeric code of î is: 140
The numeric code of ì is: 141
The numeric code of Ä is: 142
The numeric code of Å is: 143
The numeric code of É is: 144
The numeric code of æ is: 145
The numeric code of Æ is: 146
The numeric code of ô is: 147
The numeric code of ö is: 148
The numeric code of ò is: 149
The numeric code of û is: 150
Configuration: mingw2.95 - CUI Debug, Builder Type: MinGW (
Checking file dependency


C-Free 4.0 - [C:\Programmi\C-Free Standard\samples\4-1a.c]
File Edit Search View Project Build (Pro)Debug Tools Window Help
[D] mingw2.95
4-1a.c
1 /* 4.1b Print the numeric code value of all characters */
2
3 #include <stdio.h>
4
5 int main(void)
6 {
7 char ch;
8 int I;
9 for(I=0; I<=255; I++)
10 {
11
12 ch=I;
13 printf("The numeric code of %c is: %d \n", ch, ch);
14 }
15 exit(0);
16 }
"C:\Programmi\C-Free Standard\samples\4-1a.exe"
The numeric code of q is: 113
The numeric code of r is: 114
The numeric code of s is: 115
The numeric code of t is: 116
The numeric code of u is: 117
The numeric code of v is: 118
The numeric code of w is: 119
The numeric code of x is: 120
The numeric code of y is: 121
The numeric code of z is: 122
The numeric code of { is: 123
The numeric code of ! is: 124
The numeric code of } is: 125
The numeric code of ~ is: 126
The numeric code of ⌂ is: 127
The numeric code of Ç is: -128
The numeric code of ü is: -127
The numeric code of é is: -126
The numeric code of â is: -125
The numeric code of ä is: -124
The numeric code of à is: -123
The numeric code of å is: -122
The numeric code of ç is: -121
The numeric code of ê is: -120
The numeric code of ë is: -119
The numeric code of è is: -118
The numeric code of ï is: -117
The numeric code of î is: -116
The numeric code of ì is: -115
The numeric code of Ä is: -114
The numeric code of Å is: -113
The numeric code of É is: -112
The numeric code of æ is: -111


C-Free 4.0 - [C:\Programmi\C-Free Standard\samples\4-1b.c]
File Edit Search View Project Build (Pro)Debug Tools Window Help
4-2.C
4-1b.c
1 /* 4.1b Print the numeric code value of all characters */
2
3 #include <stdio.h>
4
5 int main(void)
6 {
7 char ch;
8 unsigned short int I;
9 for(I=0; I<=255; I++)
10 {
11
12 ch=I;
13 printf("The numeric codes of %c are: %d %o %x\n", ch, I,I,I);
14 }
15 exit(0);
16 }
"C:\Programmi\C-Free Standard\samples\4-1b.exe"
The numeric codes of C are: 67 103 43
The numeric codes of D are: 68 104 44
The numeric codes of E are: 69 105 45
The numeric codes of F are: 70 106 46
The numeric codes of G are: 71 107 47
The numeric codes of H are: 72 110 48
The numeric codes of I are: 73 111 49
The numeric codes of J are: 74 112 4a
The numeric codes of K are: 75 113 4b
The numeric codes of L are: 76 114 4c
The numeric codes of M are: 77 115 4d
The numeric codes of N are: 78 116 4e
The numeric codes of O are: 79 117 4f
The numeric codes of P are: 80 120 50
The numeric codes of Q are: 81 121 51
The numeric codes of R are: 82 122 52
The numeric codes of S are: 83 123 53
The numeric codes of T are: 84 124 54
The numeric codes of U are: 85 125 55
The numeric codes of V are: 86 126 56
The numeric codes of W are: 87 127 57
The numeric codes of X are: 88 130 58
The numeric codes of Y are: 89 131 59
The numeric codes of Z are: 90 132 5a
The numeric codes of [are: 91 133 5b
The numeric codes of \ are: 92 134 5c
The numeric codes of] are: 93 135 5d
The numeric codes of ^ are: 94 136 5e
The numeric codes of _ are: 95 137 5f
The numeric codes of ` are: 96 140 60
The numeric codes of a are: 97 141 61
The numeric codes of b are: 98 142 62
The numeric codes of c are: 99 143 63
The numeric codes of d are: 100 144 64
The numeric codes of e are: 101 145 65
The numeric codes of f are: 102 146 66
The numeric codes of g are: 103 147 67
The numeric codes of h are: 104 150 68
The numeric codes of i are: 105 151 69
The numeric codes of j are: 106 152 6a
The numeric codes of k are: 107 153 6b
The numeric codes of l are: 108 154 6c