Cycle: Année Préparatoire (AP)

Niveau: 2ème année (AP2)

ENSAH

A.U: 2020/2021

Semestre : Autonome

Module : Informatique 2

Pr. Ahmad EL ALLAOUI

Description du Module

- ***Généralités sur le langage C.***
- ***Types de données en langage C***
  - ✓ Instructions élémentaires. Types de variables. Instructions des entrées-sorties.
- ***Opérateurs et expressions en langage C.***
- ***Les entrées sorties conversationnelles en langage C.***
- ***Structures et instructions de contrôle en langage C.***
  - ✓ Structures de choix simple (IF ... ELSE ...). Structures à choix multiples (SWITCH …). Boucles (WHILE …, DO … WHILE, FOR …).
- ***Programmation modulaire.***
  - ✓ Fonctions. Passage de paramètres par valeur et par adresse. Variables globales et variables locales.
- ***Tableaux et pointeurs.***
  - ✓ Tableaux (cas d'une seule dimension et de plusieurs dimensions). Pointeurs. Chaines de caractères.
  - ✓ Exposition des principales méthodes prédéfinies de la bibliothèque string.h. Exploitation de ces méthodes pour résoudre des problèmes sur des données complexes.
- ***Introduction à : l'allocation dynamique de la mémoire, les structures et les fichiers.***
  - ✓ Gestion dynamique de la mémoire. Structures. Fichiers (création, suppression, différents types d'ouverture et fermeture d'un fichier et enregistrement dans un fichier)..

Evaluation du module

- Présence: plus de 3 absences en TP=>note zéro
- CC = Contrôle continu 25%, googleMeet, Zoom
- TP= 25%,
- EX = Examen 50%. Evalbox

Note du module = 0,25*CC + 0,25*TP + 0,5*EX

Objectifs

- Être capable de bien programmer
- Comprendre les différentes constructions de la programmation en C
- Savoir programmer de manière modulaire

Chapitre 1: Généralités sur le langage C

1. Généralités sur le langage C

1.1. Historique

- C a été inventé aux «Bells Laboratories » en 1972 par **Dennis M.Ritchie** avec l'objectif d'écrire un système d'exploitation(UNIX).

- En 1978 publication de livre «The C Programming Language ». par **Brian W. Kernighan** et **Dennis M.Ritchie**.

- En 1983, l'organisme ANSI (American National Standards Institute) commence le processus de normalisation du langage C. Le résultat était le standard ANSI-C.

- En 1988: deuxième édition du livre «The C Programming Language », qui respecte le standard ANSI-C. Elle est devenue la référence des programmes en C.

1. Généralités sur le langage C

1. 2. Caractéristiques du Langage C

- **Structuré:** traiter les tâches d'un programme en les mettant dans des blocs.
- **Efficace:** Possède les mêmes possibilités de contrôle de la machine que l'assembleur et il génère un **code compact et rapide.**
- **Modulaire**: Permet de découper une application en modules qui peuvent être compilés séparément.
- **Souple:** Hormis la syntaxe, peu de vérifications et d'interdits ce qui peu poser des problèmes.
- **Portable:** Permet d'utiliser le même code source sur d'autres types de machines simplement en le recompilant.
- **Extensible:** Animé par des bibliothèques de fonctions qui enrichissent le langage.

1. Généralités sur le langage C

1.3. Code source, objet et exécutable

❑ Un programme écrit en langage C forme un texte qu'on nomme **programme ou code source**, qui peut être formé de plusieurs fichiers sources.

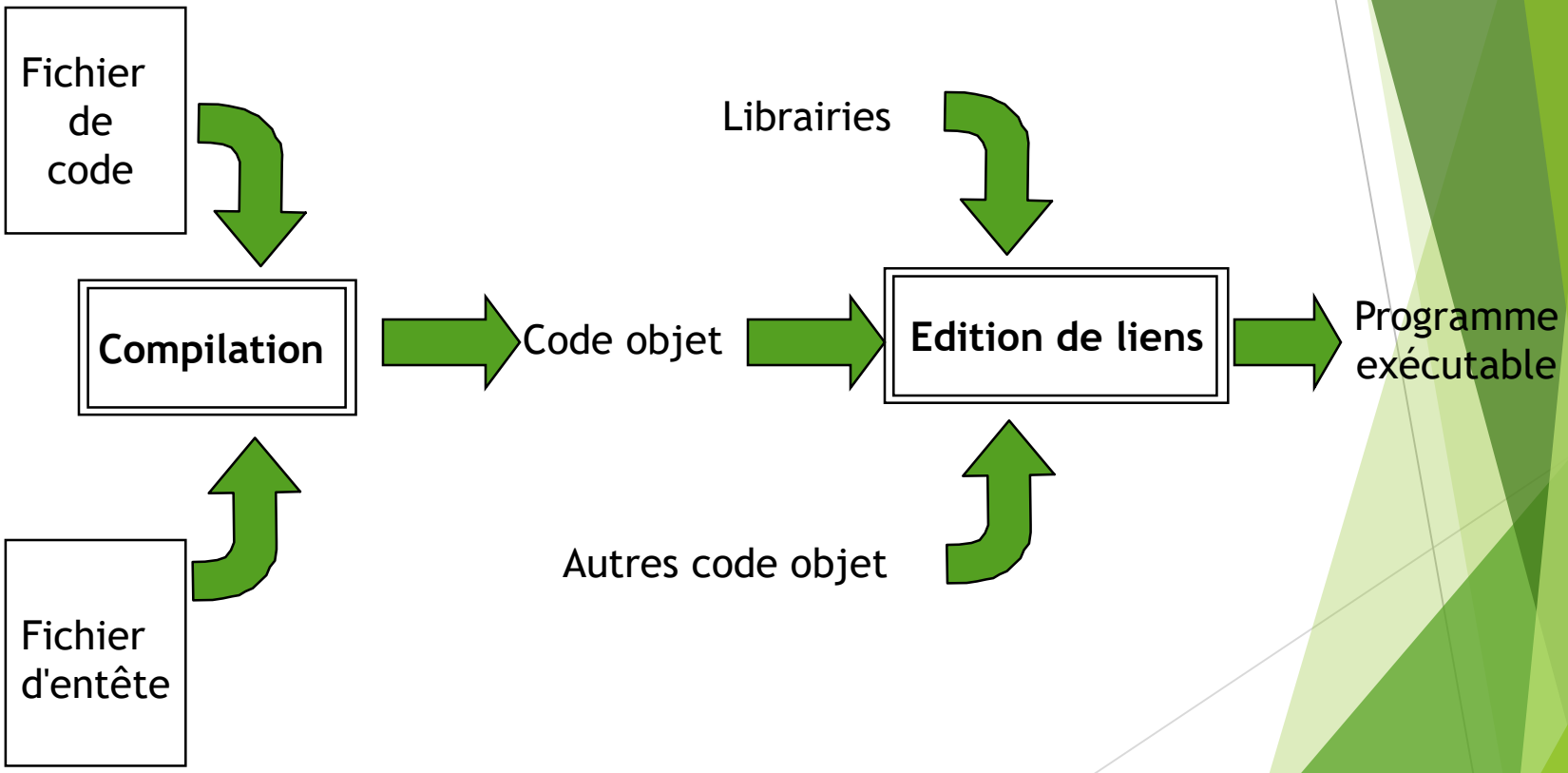
❑ Chaque fichier source est traduit par le compilateur pour obtenir un **fichier ou module objet** (formé d'instructions machine).

❑ L'éditeur de liens réunit les différents modules objets et les fonctions de la bibliothèque standard afin de former un **programme exécutable**.

1. Généralités sur le langage C

1.3. Code source, objet et exécutable

Etapes qui ont lieu avant l'exécution pour un langage compilé comme C



1. Généralités sur le langage C

Indenter = lisibilité

Prenez l'habitude de respecter (au moins au début) les règles :

- une accolade est seule sur sa ligne,
- { est alignée sur le caractère de gauche de la ligne précédente,
- } est alignée avec l'accolade ouvrante correspondante,
- après { , on commence à écrire deux caractères plus à droite.

```
#include <Lib1.h>
#include <Lib2.h>
#define X 0;
 int fonc1(int x);
 float fonc2(char a);

 int main()
   {  /*main*/
            instruction;
            instruction;
          {
instruction;
               {
instruction;
               }
          }
          instruction;
```

1. Généralités sur le langage C

1.4. Préprocesseur

- Le préprocesseur effectue un prétraitement du programme source avant qu'il soit compilé.
- Ce préprocesseur exécute des instructions particulières appelées **directives.**
- Ces directives sont identifiées par le caractère **#** en tête.

Inclusion de fichiers

```
#include <nom-de-fichier>      /* répertoire standard */

#include "nom-de-fichier"      /* répertoire courant */
```

La gestion des fichiers (**stdio.h**) /* Entrees-sorties standard */
Les fonctions mathématiques (**math.h**)
Taille des type entiers (**limits.h**)
Limites des type réels (**float.h**)
Traitement de chaînes de caractères (**string.h**)
Le traitement de caractères (**ctype.h**)
Utilitaires généraux (**stdlib.h**)
Date et heure (**time.h**)

1. Généralités sur le langage C

1.4. Compilateurs C

❑ Pour pouvoir écrire des programmes en C, vous avez besoin d'un compilateur C sur votre machine.

❑ Il existe plusieurs compilateurs respectant le standard ANSI-C. Une bonne liste est disponible sur :

cpp.developpez.com/telecharger/index/categorie/70/Compilateurs

❑ Nous allons utiliser l'environnement de développement CodeBlocks, ou Dev C++ avec le système d'exploitation Windows.

❑ Pour compiler on utilise la commande **Compile** (compiler) de l'éditeur des programmes sources utilisé. Et pour exécuter on utilise la commande **Run** (exécuter).

1. Généralités sur le langage C

1.5. Composantes d'un programme C(1)

- Directives du préprocesseur
  - ✓Inclusion des fichiers d'en-tête (fichiers avec extension .h)
  - ✓Définitions des constantes avec **#define**

- Déclaration des variables globales
- Définition des fonctions (En C, le programme principal et les sous programmes sont définis comme fonctions)
- Les commentaires : texte ignoré par le compilateur, destiné à améliorer la compréhension du code /*…..*/

1. Généralités sur le langage C

1.5. Composantes d'un programme C(2)

[*directives au préprocesseur]*

[*déclarations de variables globales]*

[*fonctions secondaires]*

main()

{

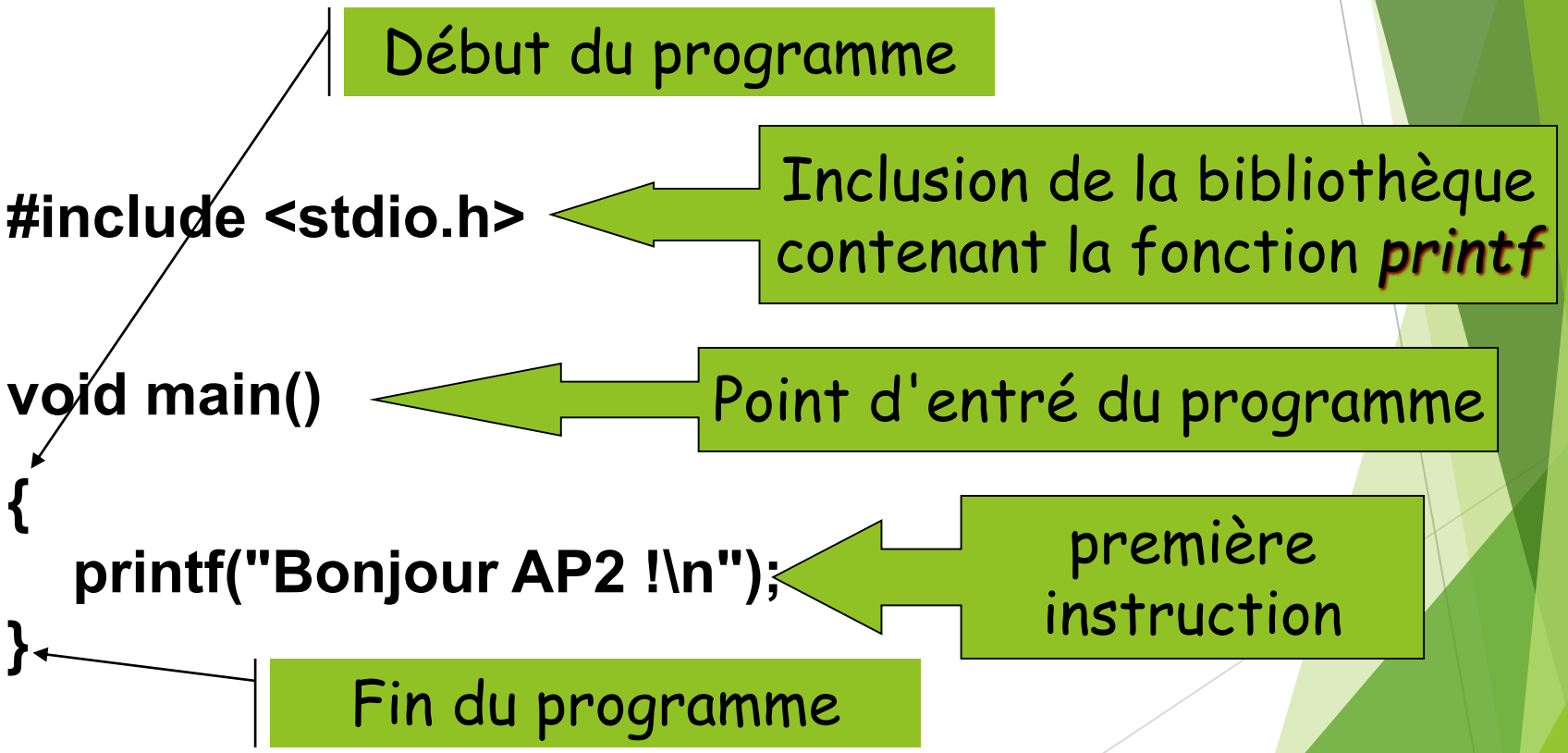
déclarations de variables locales

instructions

}

1. Généralités sur le langage C

1.5. Composantes d'un programme C(3)



1. Généralités sur le langage C

1.5. Composantes d'un programme C(3)

Exemple :

```
#include<stdio.h>
void main()
{
printf( « 1er programme C \n");
/*ceci est un commentaire*/
}
```

- *#include<stdio.h> informe le compilateur d'inclure le fichier stdio.h qui* contient les fonctions d'entrées-sorties dont la fonction printf et scanf
- La fonction **main est la fonction principale des programmes en C:** Elle se trouve obligatoirement dans tous les programmes. L'exécution d'un programme entraîne automatiquement l'appel de la fonction **main.**
- L'appel de printf avec l'argument « 1er programme C\n" permet d'afficher : 1er programme C et \n ordonne le passage à la ligne suivante
- En C, *toute instruction simple est terminée par un point-virgule ;*
- Un commentaire en C est compris entre // et la fin de la ligne ou bien entre /* et */

Chapitre 2: Types de données en C

2. Types de données en C

2.1. Instructions élémentaires(Variables)

- Une donnée est de nature **variable** ou **constante.**
- Une variable désigne un emplacement mémoire dont le contenu peut changer au cours d'un programme;
- Les variables servent à stocker les valeurs des données utilisées pendant l'exécution d'un programme.
- Les variables doivent être **déclarées avant d'être** utilisées, elles doivent être caractérisées par :
  - ✓ Un nom (**Identificateur)**
  - ✓ Un **type (entier, réel, …)**

2. Types de données en C

2.1. Instructions élémentaires(Variables: Identificateur)

Le choix d'un identificateur (nom d'une variable ou d'une fonction) est soumis à quelques règles :

- Doit être constitué uniquement de lettres, de chiffres et du caractère souligné _ (Eviter les caractères de ponctuation et les espaces)

correct: VAR_HT, VARHT **incorrect: VAR-HT, VAR HT, VAR.HT**

- Doit commencer par une lettre (y compris le caractère souligné)

correct : VAR1, _VAR1 **incorrect: 1VAR**

- Doit être différent des mots réservés du langage(L'ANSI C compte 32 mots clefs) :

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

RQ: C distingue les minuscules, des majuscules(sensible à la casse). A # a

2. Types de données en C

2.2.Types de variables.

- Le type d'une variable détermine l'ensemble des valeurs qu'elle peut prendre et le nombre d'octets à lui réserver en mémoire.

- En langage C, il n'y a que deux types de base **les entiers** et **les réels** avec différentes variantes pour chaque type.

2. Types de données en C

2.2.Types de variables(Types Entier)

- Le langage C distingue plusieurs types d'entiers:

Type	Taille	Borne inférieure	Borne supérieure
char	**1 octet(8bits)**	**$-(2^7-1) = -127$**	**$2^7-1 = +127$**
unsigned char	**1 octet**	**0**	**$2^8-1 = +255$**
short	**2 octets**	**$-(2^{15}-1) = -32\,767$**	**$2^{15}-1 = +32\,767$**
unsigned short	**2 octets**	**0**	**$2^{16}-1 = +65\,535$**
int	**4 octets**	**$-(2^{31}-1) = -2\,147\,483\,647$**	**$2^{31}-1 = +2\,147\,483\,647$**
unsigned int	**4 octets**	**0**	**$2^{32}-1 = +4\,294\,967\,295$**
long	**4 octets**	**$-(2^{31}-1) = -2\,147\,483\,647$**	**$2^{31}-1 = +2\,147\,483\,647$**
unsigned long	**4 octets**	**0**	**$2^{32}-1 = +4\,294\,967\,295$**

RQ: le type **char** est un cas particulier du type entier

2. Types de données en C

2.2.Types de variables(Types Réel).

- 3 variantes de réels :

✓**float :** réel simple précision codé sur 4 octets de **$-3.4*10^{38}$ à $3.4*10^{38}$**

✓**double :** réel double précision codé sur 8 octets de **$-1.7*10^{308}$ à $1.7*10^{308}$**

✓ **long double :** réel très grande précision codé sur 10 octets de **$-3.4*10^{4932}$ à $3.4*10^{4932}$**

2. Types de données en C

2.3. Déclaration des variables.

- Les déclarations introduisent les variables qui seront utilisées, fixent leur type et parfois aussi leur valeur de départ (initialisation)
- Syntaxe de déclaration en C

<Type> <NomVar1>,<NomVar2>,...,<NomVarN> ;

- Exemple:

```
int i, j, k;
short compteur;
char c='A';
float x, y ;
double z=1.5; // déclaration et initialisation
float r=65.2, b= 5.6;
```

2. Types de données en C

2.4. Déclaration des constantes

❑ Une constante conserve sa valeur pendant toute l'exécution d'un programme

❑ En C, on associe une valeur à une constante en utilisant :

✓ la directive ***#define*** : ***#define nom_constante valeur***

Ici la constante ne possède pas de type.

exemple: ***#define*** P_i *3.141592*

✓ le mot clé ***const*** : **const *type nom* = *expression* ;**

Dans cette instruction la constante est typée

exemple : ***const float*** P_i *=3.141592 ;*

2. Types de données en C

2.4. Déclaration des constantes

Dans un programme C, on peut manipuler 3 types de constantes :

1)constantes entières, 2)constantes réelles, 3)constantes caractères et chaînes de caractères

2.4.1. Les constantes entières

- sous forme décimale : 100, 255.
- sous forme octale, en faisant précéder le nombre par le caractère 0 (zéro) : 0144, 0377.
- sous forme hexadécimale, en faisant précéder le nombre par 0x ou 0X : 0x64, 0Xff

2. Types de données en C

2.4. Déclaration des constantes

2.4.1. Les constantes entières

- Le type attributs à une constante est automatique (C choisit la solution la plus économique)
- On peut forcer l'ordinateur à attribuer un type de notre choix à la constante en utilisant les suffixes suivants:

Suffixe	Type	Exemple
U ou u	unsigned int	245u
L ou l	long	12435L
UL ou ul	unsigned long	13456ul

2.4.2. Les constantes réelles

- La notation décimale doit comporter obligatoirement un point (correspondant à notre virgule). 10.63 -0.27 -.38 7. .29
- La notation exponentielle utilise la lettre e (ou E) pour introduire un exposant entier (puissance de 10), avec ou sans signe.

6.31E4 6.31e+4 63.1E3
24.27E-32 242.7E-33 2427e-34
87e12 87.e12 87.0E12

2. Types de données en C

2.4. Déclaration des constantes

2.4.2. Les constantes réelles

RQ: Par défaut les constantes réelles sont de type double

- ❑ Le suffixe f ou F pour forcer l'utilisation de float
- ❑ Le suffixe l ou L pour forcer l'utilisation de long double

2.4.3. Les constantes caractères

Sont toujours indiqués entre apostrophes ' '; Exemple : 'a'; 'b' ; 'A' ; '+' ; ',' ;

❑Le **caractère** 'b' a pour valeur 98 (son code ASCII). Le **caractère** 257 a pour valeur 1 (ce nombre s'écrit sur 9 bits, le bit de poids fort est perdu).

❑L'expression : 'a' + '?' vaut 160 (code ASCII de 'a' = 97 et celui de '?' = 63)

❑ char c; c = 'A'; est équivalent à char c = 'A';

int i; i = 50; est équivalent à int i = 50;

2. Types de données en C

2.4. Déclaration des constantes

2.4.3. Les constantes caractères

❑Certains caractères non imprimables possèdent une représentation conventionnelle utilisant le caractère « \ », nommé « antislash » (en anglais« back-slash »),

NOTATION EN C	CODE ASCII (hexadécimal)	ABRÉVIATION USUELLE	SIGNIFICATION
\a	07	BEL	cloche ou bip (alert ou audible bell)
\b	08	BS	Retour arrière (Backspace)
\f	0C	FF	Saut de page (Form Feed)
\n	0A	LF	Saut de ligne (Line Feed)
\r	0D	CR	Retour chariot (Carriage Return)
\t	09	HT	Tabulation horizontale (Horizontal Tab)
\v	0B	VT	Tabulation verticale (Vertical Tab)
\\	5C	\	
\'	2C	'	
\"	22	"	
\?	3F	?	

Réf: Claude Delannoy

Table de code ASCII

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com