Data

- int
- float
- double

Expressions

- I/O expressions
- Arithmetic expressions

Control Flow

- Sequential

Data

- int
- float
- double
- char

Expressions

- I/O expressions
- Arithmetic expressions

Control Flow

- Sequential

The `char` Data Type

The `char` Data Type

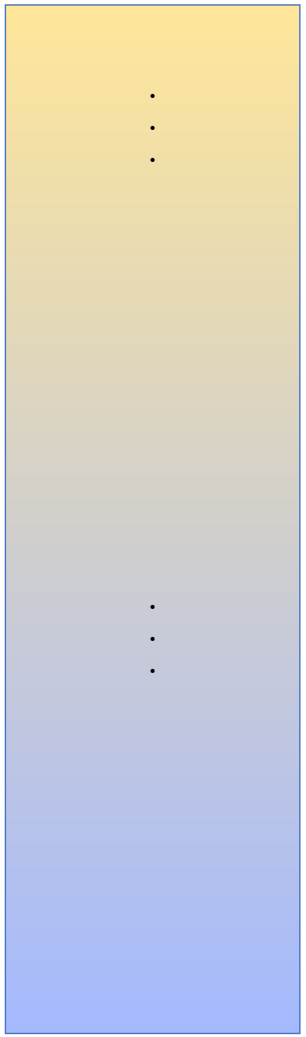
Kind of data: Characters

The `char` Data Type

Kind of data: Characters

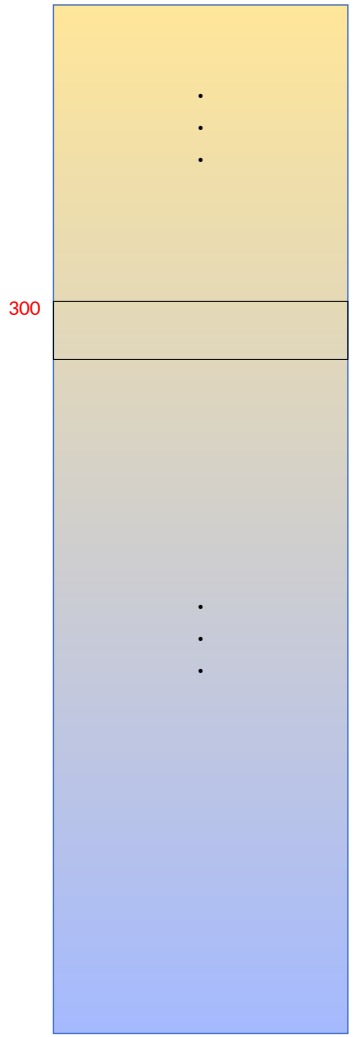
Inner representation:

‘a’



Memory

‘a’



Memory

The `char` Data Type

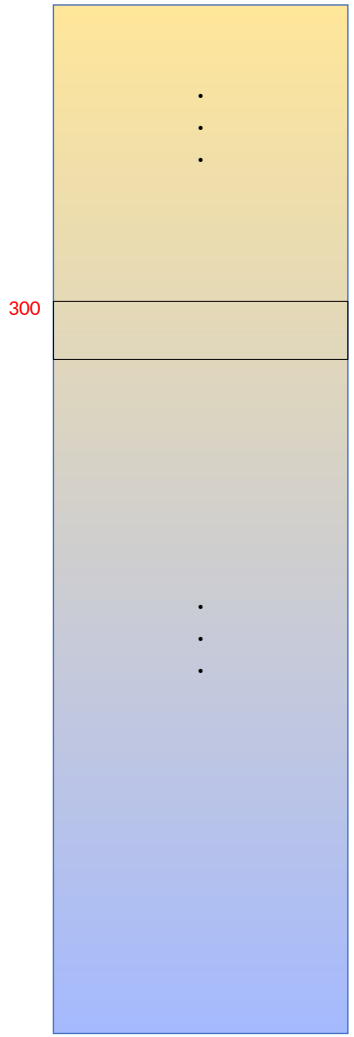
Kind of data: Characters

Inner representation:

- Each `char` data uses 1 byte (8 bits)

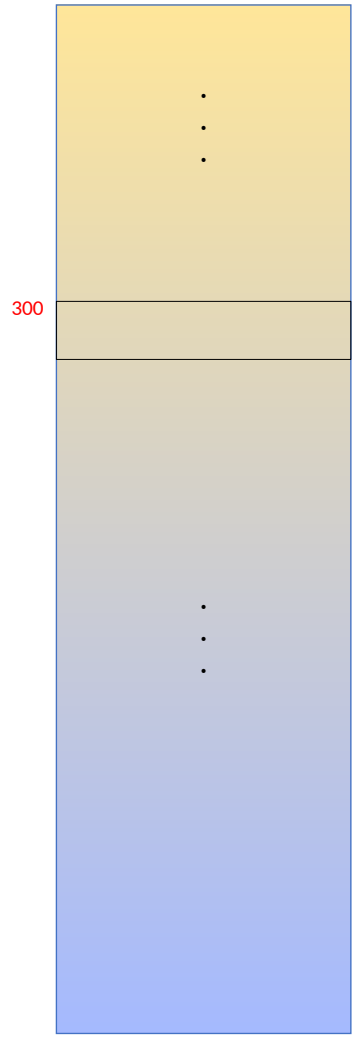
Decimal	Hex	ASCII	Decimal	Hex	ASCII	Decimal	Hex	ASCII	Decimal	Hex	ASCII
0	00	NUL	32	20	(blank)	64	40	@	96	60	`
1	01	SOH	33	21	!	65	41	A	97	61	a
2	02	STX	34	22	"	66	42	B	98	62	b
3	03	ETX	35	23	#	67	43	C	99	63	c
4	04	EOT	36	24	$	68	44	D	100	64	d
5	05	ENQ	37	25	%	69	45	E	101	65	e
6	06	ACK	38	26	&	70	46	F	102	66	f
7	07	BEL	39	27	'	71	47	G	103	67	g
8	08	BS	40	28	(	72	48	H	104	68	h
9	09	HT	41	29	)	73	49	I	105	69	i
10	0A	LF	42	2A	*	74	4A	J	106	6A	j
11	0B	VT	43	2B	+	75	4B	K	107	6B	k
12	0C	FF	44	2C	,	76	4C	L	108	6C	l
13	0D	CR	45	2D	-	77	4D	M	109	6D	m
14	0E	SO	46	2E	.	78	4E	N	110	6E	n
15	0F	SI	47	2F	/	79	4F	O	111	6F	o
16	10	DLE	48	30	0	80	50	P	112	70	p
17	11	DC1	49	31	1	81	51	Q	113	71	q
18	12	DC2	50	32	2	82	52	R	114	72	r
19	13	DC3	51	33	3	83	53	S	115	73	s
20	14	DC4	52	34	4	84	54	T	116	74	t
21	15	NAK	53	35	5	85	55	U	117	75	u
22	16	SYN	54	36	6	86	56	V	118	76	v
23	17	ETB	55	37	7	87	57	W	119	77	w
24	18	CAN	56	38	8	88	58	X	120	78	x
25	19	EM	57	39	9	89	59	Y	121	79	y
26	1A	SUB	58	3A	:	90	5A	Z	122	7A	z
27	1B	ESC	59	3B	;	91	5B	[	123	7B	{
28	1C	FS	60	3C	<	92	5C	\	124	7C	|
29	1D	GS	61	3D	=	93	5D	]	125	7D	}
30	1E	RS	62	3E	>	94	5E	^	126	7E	~
31	1F	US	63	3F	?	95	5F	_	127	7F	(delete)

‘a’



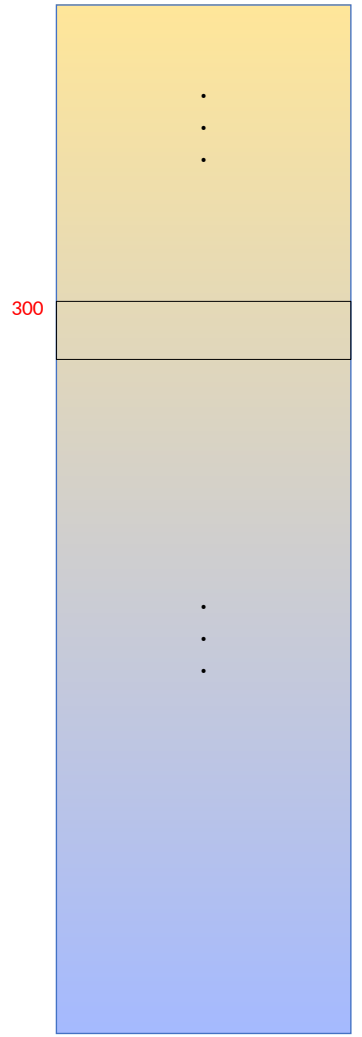
Memory

‘a’ ASCII Value 97



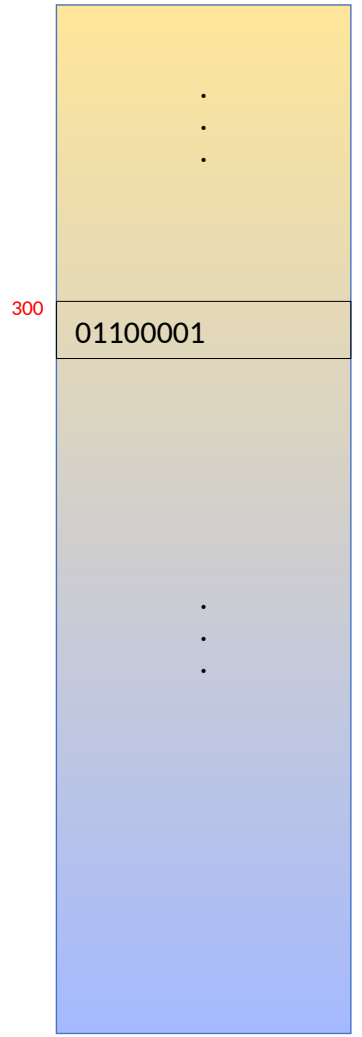
‘a’ ASCII Value 97

$(97)_{10} = (01100001)_2$



‘a’ ASCII Value 97

$(97)_{10} = (01100001)_2$



The `char` Data Type

Kind of data: Characters

Inner representation:

- Each `char` data uses 1 byte (8 bits)
- The characters are mapped to numbers by the ASCII table, which are then represented in binary

What’s My ASCII Value?

Write a program that reads from the user a single character, and prints it’s ASCII value.

What’s My ASCII Value?

Write a program that reads from the user a single character, and prints it’s ASCII value.

Example

Please enter a character:

What's My ASCII Value?

Write a program that reads from the user a single character, and prints it's ASCII value.

Example

Please enter a character:

T

What's My ASCII Value?

Write a program that reads from the user a single character, and prints it's ASCII value.

Example

Please enter a character:

T

The ASCII value of T is 84

The `char` Data Type

Kind of data: Characters

Inner representation:

- Each `char` data uses 1 byte (8 bits)
- The characters are mapped to numbers by the ASCII table, which are then represented in binary

C++ literals:

char Literals

```
int main(){
        char ch;









        return 0;
}
```

char Literals

```
int main(){
        char ch;

        ch = a;









        return 0;
}
```

char Literals

```
int main(){
        char ch;

        ch = a;

        return 0;
}
```

The `char` Data Type

Kind of data: Characters

Inner representation:

- Each `char` data uses 1 byte (8 bits)
- The characters are mapped to numbers by the ASCII table, which are then represented in binary

C++ literals:

The `char` Data Type

Kind of data: Characters

Inner representation:

- Each `char` data uses 1 byte (8 bits)
- The characters are mapped to numbers by the ASCII table, which are then represented in binary

C++ literals: ‘a’, ‘B’, ‘3’, ‘$’

char Literals

```
int main(){
        char ch;

        ch = 'a';









        return 0;
}
```

char Literals

```
int main(){
        char ch;

        ch = 'a';
        cout<<ch<<endl;









        return 0;
}
```

char Literals

```
int main(){
        char ch;

        ch = 'a';
        cout<<ch<<endl;

        cout<<'b'<<endl;




        return 0;
}
```

char Literals

```
int main(){
        char ch;

        ch = 'a';
        cout<<ch<<endl;

        cout<<'b'<<endl;


        ch = "a";


        return 0;
}
```

char Literals

```
int main(){
        char ch;

        ch = 'a';
        cout<<ch<<endl;

        cout<<'b'<<endl;


        ch = "a";


        return 0;
}
```

The `char` Data Type

Kind of data: Characters

Inner representation:

- Each `char` data uses 1 byte (8 bits)
- The characters are mapped to numbers by the ASCII table, which are then represented in binary

C++ literals: ‘a’, ‘B’, ‘3’, ‘$’

The `char` Data Type

Kind of data: Characters

Inner representation:

- Each `char` data uses 1 byte (8 bits)
- The characters are mapped to numbers by the ASCII table, which are then represented in binary

C++ literals: ‘a’, ‘B’, ‘3’, ‘$’, ‘\n’

Escape Characters

```
int main(){
        char ch;









        return 0;
}
```

Escape Characters

```
int main(){
        char ch;


        cout<<'\n';









        return 0;
}
```

Escape Characters

```
int main(){
        char ch;


        cout<<'\n';
        cout<<endl;









        return 0;
}
```

Escape Characters

```
int main(){
        char ch;


        cout<<'\n';
        cout<<endl;

        ch = '\n';
        cout<<ch;









        return 0;
}
```

Escape Characters

```
int main(){
        char ch;


        cout<<'\n';
        cout<<endl;

        ch = '\n';
        cout<<ch;


        cout<<"abc"<<'\n';




        return 0;
}
```

Escape Characters

```
int main(){
        char ch;


        cout<<'\n';
        cout<<endl;

        ch = '\n';
        cout<<ch;


        cout<<"abc"<<'\n';
        cout<<"abc\n";



        return 0;
}
```

The `char` Data Type

Kind of data: Characters

Inner representation:

- Each `char` data uses 1 byte (8 bits)
- The characters are mapped to numbers by the ASCII table, which are then represented in binary

C++ literals: ‘a’, ‘B’, ‘3’, ‘$’, ‘\n’

The `char` Data Type

Kind of data: Characters

Inner representation:

- Each `char` data uses 1 byte (8 bits)
- The characters are mapped to numbers by the ASCII table, which are then represented in binary

C++ literals: ‘a’, ‘B’, ‘3’, ‘$’, ‘\n’, ‘\t’, ‘\\’, …

The `char` Data Type

Kind of data: Characters

Inner representation:

- Each `char` data uses 1 byte (8 bits)
- The characters are mapped to numbers by the ASCII table, which are then represented in binary

C++ literals: ‘a’, ‘B’, ‘3’, ‘$’, ‘\n’, ‘\t’, ‘\\’, …

Arithmetic Operators:

The `char` Data Type

Kind of data: Characters

Inner representation:

- Each `char` data uses 1 byte (8 bits)
- The characters are mapped to numbers by the ASCII table, which are then represented in binary

C++ literals: ‘a’, ‘B’, ‘3’, ‘$’, ‘\n’, ‘\t’, ‘\\’, …

Arithmetic Operators: +

Decimal	Hex	ASCII	Decimal	Hex	ASCII	Decimal	Hex	ASCII	Decimal	Hex	ASCII
0	00	NUL	32	20	(blank)	64	40	@	96	60	`
1	01	SOH	33	21	!	65	41	A	97	61	a
2	02	STX	34	22	"	66	42	B	98	62	b
3	03	ETX	35	23	#	67	43	C	99	63	c
4	04	EOT	36	24	$	68	44	D	100	64	d
5	05	ENQ	37	25	%	69	45	E	101	65	e
6	06	ACK	38	26	&	70	46	F	102	66	f
7	07	BEL	39	27	'	71	47	G	103	67	g
8	08	BS	40	28	(	72	48	H	104	68	h
9	09	HT	41	29	)	73	49	I	105	69	i
10	0A	LF	42	2A	*	74	4A	J	106	6A	j
11	0B	VT	43	2B	+	75	4B	K	107	6B	k
12	0C	FF	44	2C	,	76	4C	L	108	6C	l
13	0D	CR	45	2D	-	77	4D	M	109	6D	m
14	0E	SO	46	2E	.	78	4E	N	110	6E	n
15	0F	SI	47	2F	/	79	4F	O	111	6F	o
16	10	DLE	48	30	0	80	50	P	112	70	p
17	11	DC1	49	31	1	81	51	Q	113	71	q
18	12	DC2	50	32	2	82	52	R	114	72	r
19	13	DC3	51	33	3	83	53	S	115	73	s
20	14	DC4	52	34	4	84	54	T	116	74	t
21	15	NAK	53	35	5	85	55	U	117	75	u
22	16	SYN	54	36	6	86	56	V	118	76	v
23	17	ETB	55	37	7	87	57	W	119	77	w
24	18	CAN	56	38	8	88	58	X	120	78	x
25	19	EM	57	39	9	89	59	Y	121	79	y
26	1A	SUB	58	3A	:	90	5A	Z	122	7A	z
27	1B	ESC	59	3B	;	91	5B	[	123	7B	{
28	1C	FS	60	3C	<	92	5C	\	124	7C	|
29	1D	GS	61	3D	=	93	5D	]	125	7D	}
30	1E	RS	62	3E	>	94	5E	^	126	7E	~
31	1F	US	63	3F	?	95	5F	_	127	7F	(delete)

Decimal	Hex	ASCII	Decimal	Hex	ASCII	Decimal	Hex	ASCII	Decimal	Hex	ASCII
0	00	NUL	32	20	(blank)	64	40	@	96	60	`
1	01	SOH	33	21	!	65	41	A	97	61	a
2	02	STX	34	22	"	66	42	B	98	62	b
3	03	ETX	35	23	#	67	43	C	99	63	c
4	04	EOT	36	24	$	68	44	D	100	64	d
5	05	ENQ	37	25	%	69	45	E	101	65	e
6	06	ACK	38	26	&	70	46	F	102	66	f
7	07	BEL	39	27	'	71	47	G	103	67	g
8	08	BS	40	28	(	72	48	H	104	68	h
9	09	HT	41	29	)	73	49	I	105	69	i
10	0A	LF	42	2A	*	74	4A	J	106	6A	j
11	0B	VT	43	2B	+	75	4B	K	107	6B	k
12	0C	FF	44	2C	,	76	4C	L	108	6C	l
13	0D	CR	45	2D	-	77	4D	M	109	6D	m
14	0E	SO	46	2E	.	78	4E	N	110	6E	n
15	0F	SI	47	2F	/	79	4F	O	111	6F	o
16	10	DLE	48	30	0	80	50	P	112	70	p
17	11	DC1	49	31	1	81	51	Q	113	71	q
18	12	DC2	50	32	2	82	52	R	114	72	r
19	13	DC3	51	33	3	83	53	S	115	73	s
20	14	DC4	52	34	4	84	54	T	116	74	t
21	15	NAK	53	35	5	85	55	U	117	75	u
22	16	SYN	54	36	6	86	56	V	118	76	v
23	17	ETB	55	37	7	87	57	W	119	77	w
24	18	CAN	56	38	8	88	58	X	120	78	x
25	19	EM	57	39	9	89	59	Y	121	79	y
26	1A	SUB	58	3A	:	90	5A	Z	122	7A	z
27	1B	ESC	59	3B	;	91	5B	[	123	7B	{
28	1C	FS	60	3C	<	92	5C	\	124	7C	|
29	1D	GS	61	3D	=	93	5D	]	125	7D	}
30	1E	RS	62	3E	>	94	5E	^	126	7E	~
31	1F	US	63	3F	?	95	5F	_	127	7F	(delete)

Arithmetic Operators for `char`

```
int main(){
        char ch1, ch2;

        ch1 = 'a';









        return 0;
}
```

Arithmetic Operators for `char`

```
int main(){
        char ch1, ch2;

        ch1 = 'a';
        ch2 = 'a' + 1;




        return 0;
}
```

Arithmetic Operators for `char`

```
int main(){
        char ch1, ch2;

        ch1 = 'a';
        ch2 = 'a' + 1;

        cout<<ch2<<endl;



        return 0;
}
```

Arithmetic Operators for `char`

```
int main(){
        char ch1, ch2;

        ch1 = 'a';
        ch2 = 'a' + 1;

        cout<<ch2<<endl;
        cout<<'a' + 1<<endl;


        return 0;
}
```

Arithmetic Operators for `char`

```
int main(){
        char ch1, ch2;

        ch1 = 'a';
        ch2 = 'a' + 1;

        cout<<ch2<<endl;
        cout<<'a' + 1<<endl;
        cout<<(char)('a' + 1)<<endl;

        return 0;
}
```

The `char` Data Type

Kind of data: Characters

Inner representation:

- Each `char` data uses 1 byte (8 bits)
- The characters are mapped to numbers by the ASCII table, which are then represented in binary

C++ literals: ‘a’, ‘B’, ‘3’, ‘$’, ‘\n’, ‘\t’, ‘\\’, …

Arithmetic Operators: +

The `char` Data Type

Kind of data: Characters

Inner representation:

- Each `char` data uses 1 byte (8 bits)
- The characters are mapped to numbers by the ASCII table, which are then represented in binary

C++ literals: ‘a’, ‘B’, ‘3’, ‘$’, ‘\n’, ‘\t’, ‘\\’, …

Arithmetic Operators: +, -, =

Convert to UPPER CASE

Write a program that reads from the user a lower case letter, and prints it’s corresponding upper case letter.

Convert to UPPER CASE

Write a program that reads from the user a lower case letter, and prints it's corresponding upper case letter.

Example

Please enter a lower case letter:

Convert to UPPER CASE

Write a program that reads from the user a lower case letter, and prints it's corresponding upper case letter.

Example

Please enter a lower case letter:

t

Convert to UPPER CASE

Write a program that reads from the user a lower case letter, and prints it's corresponding upper case letter.

Example

Please enter a lower case letter:

t

The upper case of t is T

Decimal	Hex	ASCII	Decimal	Hex	ASCII	Decimal	Hex	ASCII	Decimal	Hex	ASCII
0	00	NUL	32	20	(blank)	64	40	@	96	60	`
1	01	SOH	33	21	!	65	41	A	97	61	a
2	02	STX	34	22	"	66	42	B	98	62	b
3	03	ETX	35	23	#	67	43	C	99	63	c
4	04	EOT	36	24	$	68	44	D	100	64	d
5	05	ENQ	37	25	%	69	45	E	101	65	e
6	06	ACK	38	26	&	70	46	F	102	66	f
7	07	BEL	39	27	'	71	47	G	103	67	g
8	08	BS	40	28	(	72	48	H	104	68	h
9	09	HT	41	29	)	73	49	I	105	69	i
10	0A	LF	42	2A	*	74	4A	J	106	6A	j
11	0B	VT	43	2B	+	75	4B	K	107	6B	k
12	0C	FF	44	2C	,	76	4C	L	108	6C	l
13	0D	CR	45	2D	-	77	4D	M	109	6D	m
14	0E	SO	46	2E	.	78	4E	N	110	6E	n
15	0F	SI	47	2F	/	79	4F	O	111	6F	o
16	10	DLE	48	30	0	80	50	P	112	70	p
17	11	DC1	49	31	1	81	51	Q	113	71	q
18	12	DC2	50	32	2	82	52	R	114	72	r
19	13	DC3	51	33	3	83	53	S	115	73	s
20	14	DC4	52	34	4	84	54	T	116	74	t
21	15	NAK	53	35	5	85	55	U	117	75	u
22	16	SYN	54	36	6	86	56	V	118	76	v
23	17	ETB	55	37	7	87	57	W	119	77	w
24	18	CAN	56	38	8	88	58	X	120	78	x
25	19	EM	57	39	9	89	59	Y	121	79	y
26	1A	SUB	58	3A	:	90	5A	Z	122	7A	z
27	1B	ESC	59	3B	;	91	5B	[	123	7B	{
28	1C	FS	60	3C	<	92	5C	\	124	7C	|
29	1D	GS	61	3D	=	93	5D	]	125	7D	}
30	1E	RS	62	3E	>	94	5E	^	126	7E	~
31	1F	US	63	3F	?	95	5F	_	127	7F	(delete)

Decimal	Hex	ASCII	Decimal	Hex	ASCII	Decimal	Hex	ASCII	Decimal	Hex	ASCII
0	00	NUL	32	20	(blank)	64	40	@	96	60	`
1	01	SOH	33	21	!	65	41	A	97	61	a
2	02	STX	34	22	"	66	42	B	98	62	b
3	03	ETX	35	23	#	67	43	C	99	63	c
4	04	EOT	36	24	$	68	44	D	100	64	d
5	05	ENQ	37	25	%	69	45	E	101	65	e
6	06	ACK	38	26	&	70	46	F	102	66	f
7	07	BEL	39	27	'	71	47	G	103	67	g
8	08	BS	40	28	(	72	48	H	104	68	h
9	09	HT	41	29	)	73	49	I	105	69	i
10	0A	LF	42	2A	*	74	4A	J	106	6A	j
11	0B	VT	43	2B	+	75	4B	K	107	6B	k
12	0C	FF	44	2C	,	76	4C	L	108	6C	l
13	0D	CR	45	2D	-	77	4D	M	109	6D	m
14	0E	SO	46	2E	.	78	4E	N	110	6E	n
15	0F	SI	47	2F	/	79	4F	O	111	6F	o
16	10	DLE	48	30	0	80	50	P	112	70	p
17	11	DC1	49	31	1	81	51	Q	113	71	q
18	12	DC2	50	32	2	82	52	R	114	72	r
19	13	DC3	51	33	3	83	53	S	115	73	s
20	14	DC4	52	34	4	84	54	T	116	74	t
21	15	NAK	53	35	5	85	55	U	117	75	u
22	16	SYN	54	36	6	86	56	V	118	76	v
23	17	ETB	55	37	7	87	57	W	119	77	w
24	18	CAN	56	38	8	88	58	X	120	78	x
25	19	EM	57	39	9	89	59	Y	121	79	y
26	1A	SUB	58	3A	:	90	5A	Z	122	7A	z
27	1B	ESC	59	3B	;	91	5B	[	123	7B	{
28	1C	FS	60	3C	<	92	5C	\	124	7C	|
29	1D	GS	61	3D	=	93	5D	]	125	7D	}
30	1E	RS	62	3E	>	94	5E	^	126	7E	~
31	1F	US	63	3F	?	95	5F	_	127	7F	(delete)

Decimal	Hex	ASCII	Decimal	Hex	ASCII	Decimal	Hex	ASCII	Decimal	Hex	ASCII
0	00	NUL	32	20	(blank)	64	40	@	96	60	`
1	01	SOH	33	21	!	65	41	A	97	61	a
2	02	STX	34	22	"	66	42	B	98	62	b
3	03	ETX	35	23	#	67	43	C	99	63	c
4	04	EOT	36	24	$	68	44	D	100	64	d
5	05	ENQ	37	25	%	69	45	E	101	65	e
6	06	ACK	38	26	&	70	46	F	102	66	f
7	07	BEL	39	27	'	71	47	G	103	67	g
8	08	BS	40	28	(	72	48	H	104	68	h
9	09	HT	41	29	)	73	49	I	105	69	i
10	0A	LF	42	2A	*	74	4A	J	106	6A	j
11	0B	VT	43	2B	+	75	4B	K	107	6B	k
12	0C	FF	44	2C	,	76	4C	L	108	6C	l
13	0D	CR	45	2D	-	77	4D	M	109	6D	m
14	0E	SO	46	2E	.	78	4E	N	110	6E	n
15	0F	SI	47	2F	/	79	4F	O	111	6F	o
16	10	DLE	48	30	0	80	50	P	112	70	p
17	11	DC1	49	31	1	81	51	Q	113	71	q
18	12	DC2	50	32	2	82	52	R	114	72	r
19	13	DC3	51	33	3	83	53	S	115	73	s
20	14	DC4	52	34	4	84	54	T	116	74	t
21	15	NAK	53	35	5	85	55	U	117	75	u
22	16	SYN	54	36	6	86	56	V	118	76	v
23	17	ETB	55	37	7	87	57	W	119	77	w
24	18	CAN	56	38	8	88	58	X	120	78	x
25	19	EM	57	39	9	89	59	Y	121	79	y
26	1A	SUB	58	3A	:	90	5A	Z	122	7A	z
27	1B	ESC	59	3B	;	91	5B	[	123	7B	{
28	1C	FS	60	3C	<	92	5C	\	124	7C	|
29	1D	GS	61	3D	=	93	5D	]	125	7D	}
30	1E	RS	62	3E	>	94	5E	^	126	7E	~
31	1F	US	63	3F	?	95	5F	_	127	7F	(delete)

Decimal	Hex	ASCII	Decimal	Hex	ASCII	Decimal	Hex	ASCII	Decimal	Hex	ASCII
0	00	NUL	32	20	(blank)	64	40	@	96	60	`
1	01	SOH	33	21	!	65	41	A	97	61	a
2	02	STX	34	22	"	66	42	B	98	62	b
3	03	ETX	35	23	#	67	43	C	99	63	c
4	04	EOT	36	24	$	68	44	D	100	64	d
5	05	ENQ	37	25	%	69	45	E	101	65	e
6	06	ACK	38	26	&	70	46	F	102	66	f
7	07	BEL	39	27	'	71	47	G	103	67	g
8	08	BS	40	28	(	72	48	H	104	68	h
9	09	HT	41	29	)	73	49	I	105	69	i
10	0A	LF	42	2A	*	74	4A	J	106	6A	j
11	0B	VT	43	2B	+	75	4B	K	107	6B	k
12	0C	FF	44	2C	,	76	4C	L	108	6C	l
13	0D	CR	45	2D	-	77	4D	M	109	6D	m
14	0E	SO	46	2E	.	78	4E	N	110	6E	n
15	0F	SI	47	2F	/	79	4F	O	111	6F	o
16	10	DLE	48	30	0	80	50	P	112	70	p
17	11	DC1	49	31	1	81	51	Q	113	71	q
18	12	DC2	50	32	2	82	52	R	114	72	r
19	13	DC3	51	33	3	83	53	S	115	73	s
20	14	DC4	52	34	4	84	54	T	116	74	t
21	15	NAK	53	35	5	85	55	U	117	75	u
22	16	SYN	54	36	6	86	56	V	118	76	v
23	17	ETB	55	37	7	87	57	W	119	77	w
24	18	CAN	56	38	8	88	58	X	120	78	x
25	19	EM	57	39	9	89	59	Y	121	79	y
26	1A	SUB	58	3A	:	90	5A	Z	122	7A	z
27	1B	ESC	59	3B	;	91	5B	[	123	7B	{
28	1C	FS	60	3C	<	92	5C	\	124	7C	|
29	1D	GS	61	3D	=	93	5D	]	125	7D	}
30	1E	RS	62	3E	>	94	5E	^	126	7E	~
31	1F	US	63	3F	?	95	5F	_	127	7F	(delete)

Data

- int
- float
- double
- char

Expressions

- I/O expressions
- Arithmetic expressions

Control Flow

- Sequential

Data

- int
- float
- double
- char
- string

Expressions

- I/O expressions
- Arithmetic expressions

Control Flow

- Sequential

The `string` Class

The `string` Class

Note: **`string`** is not a C++ built-in type. To use it you need to have: **`#include <string>`**

-

The `string` Class

Note: **`string`** is not a C++ built-in type. To use it you need to have: **`#include <string>`**

-

Kind of data:

The `string` Class

Note: **`string`** is not a C++ built-in type. To use it you need to have: **`#include <string>`**

Kind of data: Strings/Text

The `string` Class

Note: **`string`** is not a C++ built-in type. To use it you need to have: **`#include <string>`**

Kind of data: Strings/Text

Inner representation:

The `string` Class

Note: **`string`** is not a C++ built-in type. To use it you need to have: **`#include <string>`**

Kind of data: Strings/Text

Inner representation: Sequence of characters

The `string` Class

Note: **`string`** is not a C++ built-in type. To use it you need to have: **`#include <string>`**

Kind of data: Strings/Text

Inner representation: Sequence of characters

C++ literals:

The `string` Class

Note: **`string`** is not a C++ built-in type. To use it you need to have: **`#include <string>`**

Kind of data: Strings/Text

Inner representation: Sequence of characters

C++ literals: ”abc”, ”This is a string\n”, …

The `string` Class

Note: **`string`** is not a C++ built-in type. To use it you need to have: **`#include <string>`**

Kind of data: Strings/Text

Inner representation: Sequence of characters

C++ literals: ”abc”, ”This is a string\n”, …

Arithmetic Operators:

The `string` Class

Note: **`string`** is not a C++ built-in type. To use it you need to have: **`#include <string>`**

Kind of data: Strings/Text

Inner representation: Sequence of characters

C++ literals: ”abc”, ”This is a string\n”, …

Arithmetic Operators: +, =

```cpp
#include <iostream>

using namespace std;

int main(){









    return 0;
}
```

```cpp
#include <iostream>

using namespace std;

int main(){
	int x;
	double y;









	return 0;
}
```

```cpp
#include <iostream>

using namespace std;

int main(){
	int x;
	double y;


	x = 5;
	y = 7.3;









	return 0;
}
```

```cpp
#include <iostream>
#include <string>
using namespace std;

int main(){
	int x;
	double y;


	x = 5;
	y = 7.3;









	return 0;
}
```

```cpp
#include <iostream>
#include <string>
using namespace std;

int main(){
	int x;
	double y;
	string s;

	x = 5;
	y = 7.3;









	return 0;
}
```

```
#include <iostream>
#include <string>
using namespace std;

int main(){
        int x;
        double y;
        string s;

        x = 5;
        y = 7.3;
        s = ”Hello”;









        return 0;
}
```

```cpp
#include <iostream>
#include <string>
using namespace std;

int main(){
	int x;
	double y;
	string s;

	x = 5;
	y = 7.3;
	s = ”Hello”;

	cout<<s<<endl;









	return 0;
}
```

```
#include <iostream>
#include <string>
using namespace std;

int main(){
	int x;
	double y;
	string s;

	x = 5;
	y = 7.3;
	s = ”Hello”;

	cout<<s<<endl;

	cout<<s + ” world”<<endl;




	return 0;
}
```

```
#include <iostream>
#include <string>
using namespace std;

int main(){
	int x;
	double y;
	string s;

	x = 5;
	y = 7.3;
	s = ”Hello”;

	cout<<s<<endl;

	cout<<s + ” world”<<endl;

	s = s + ” world”;
	cout<<s<<endl;

	return 0;
}
```