Princeton University
COS 217: Introduction to Programming Systems
C Primitive Data Types

Type: int

Description: A (positive or negative) integer.

Size: System dependent. On FC010 with gcc217: 4 bytes.

Example Variable Declarations:

```
int iFirst;
signed int iSecond;
```

Example Literals (assuming size is 4 bytes):

C Literal	Binary Representation	Note
123	00000000 00000000 00000000 01111011	decimal form
-123	11111111 11111111 11111111 10000101	negative form
0173	00000000 00000000 00000000 01111011	octal form
0x7B	00000000 00000000 00000000 01111011	hexadecimal form
2147483647	01111111 11111111 11111111 11111111	largest
-2147483648	10000000 00000000 00000000 00000000	smallest

Type: unsigned int

Description: A non-negative integer.

Size: System dependent. sizeof(unsigned int) == sizeof(int). On FC010 with gcc217: 4 bytes.

Example Variable Declaration:

```
unsigned int uiFirst;
unsigned uiSecond;
```

Example Literals (assuming size is 4 bytes):

C Literal	Binary Representation	Note
123U	00000000 00000000 00000000 01111011	decimal form
0173U	00000000 00000000 00000000 01111011	octal form
0x7BU	00000000 00000000 00000000 01111011	hexadecimal form
4294967295U	11111111 11111111 11111111 11111111	largest
0U	00000000 00000000 00000000 00000000	smallest

Type: long

Description: A (positive or negative) integer.

Size: System dependent. sizeof(long) >= sizeof(int). On FC010 with gcc217: 8 bytes.

Example Variable Declarations:

```
long lFirst;
long int iSecond;
signed long lThird;
signed long int lFourth;
```

Example Literals (assuming size is 8 bytes):

C Literal	Binary Representation/Note
123L	00000000 00000000 00000000 00000000 00000000 00000000 00000000 01111011 decimal form
-123L	11111111 11111111 11111111 11111111 11111111 11111111 11111111 10000101 negative form
0173L	00000000 00000000 00000000 00000000 00000000 00000000 00000000 01111011 octal form
0x7BL	00000000 00000000 00000000 00000000 00000000 00000000 00000000 01111011 hexadecimal form
9223372036854775807L	01111111 11111111 11111111 11111111 11111111 11111111 11111111 11111111 largest
-9223372036854775808L	10000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 smallest

---

Type: unsigned long

Description: A non-negative integer.

Size: System dependent. sizeof(unsigned long) == sizeof(long). On FC010 with gcc217: 8 bytes.

Example Variable Declarations:

```
unsigned long ulFirst;
unsigned long int ulSecond;
```

Example Literals (assuming size is 8 bytes):

C Literal	Binary Representation/Note
123UL	00000000 00000000 00000000 00000000 00000000 00000000 00000000 01111011 decimal form
0173UL	00000000 00000000 00000000 00000000 00000000 00000000 00000000 01111011 octal form
0x7BUL	00000000 00000000 00000000 00000000 00000000 00000000 00000000 01111011 hexadecimal form
18446744073709551615UL	11111111 11111111 11111111 11111111 11111111 11111111 11111111 11111111 largest
0UL	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 smallest

---

Type: char

Description: A (positive or negative) integer. Usually represents a character according to a character code (e.g., ASCII).

Size: 1 byte.

Example Variable Declarations:

```
char cFirst;
signed char cSecond;
```

Example Literals (assuming the ASCII code is used):

C Literal	Binary Representation	Note
'a'	01100001	character form
(char)97	01100001	decimal form
(char)0141	01100001	octal form
(char)0x61	01100001	hexadecimal form
'\o141'	01100001	octal character form
'\x61'	01100001	hexadecimal character form

(char)123	01111011	decimal form
(char)-123	10000101	negative form
(char)127	01111111	largest
(char)-128	10000000	smallest
'\0'	00000000	the null character
'\a'	00000111	bell
'\b'	00001000	backspace
'\f'	00001100	formfeed
'\n'	00001010	newline
'\r'	00001101	carriage return
'\t'	00001001	horizontal tab
'\v'	00001011	vertical tab
'\\'	01011100	backslash
'\''	00100111	single quote

Type: unsigned char

Description: A non-negative integer. Usually represents a character according to a character code (e.g., ASCII).

Size: 1 byte.

Example Variable Declaration:

```
unsigned char ucFirst;
```

Example Literals (assuming the ASCII code is used):

C Literal	Binary Representation	Note
(unsigned char)'a'	01100001	character form
(unsigned char)97	01100001	decimal form
(unsigned char)255	11111111	largest
(unsigned char)0	00000000	smallest

Note: On most systems including FC010 with gcc217, "char" is the same as "signed char".
On some systems, "char" is the same as "unsigned char".

Type: short

Description: A (positive or negative) integer.

Size: System dependent. sizeof(short) <= sizeof(int). On FC010 with gcc217: 2 bytes.

Example Variable Declarations:

```
short sFirst;
short int sSecond;
signed short sThird;
signed short int sFourth;
```

Example Literals (assuming size is 2 bytes):

C Literal	Binary Representation	Note
(short)123	00000000 01111011	decimal form
(short)-123	11111111 10000101	negative form
(short)32767	01111111 11111111	largest
(short)-32768	10000000 00000000	smallest
(short)0173	00000000 01111011	octal form
(short)0x7B	00000000 01111011	hexadecimal form

Type: unsigned short

Description: A non-negative integer.

Size: System dependent. sizeof(unsigned short) == sizeof(short). On FC010 with gcc217: 2 bytes.

Example Variable Declarations:

```
unsigned short usFirst;
unsigned short int usSecond;
```

Example Literals (assuming size is 2 bytes):

C Literal	Binary Representation	Note
(unsigned short)123	00000000 01111011	decimal form
(unsigned short)0173	00000000 01111011	octal form
(unsigned short)0x7B	00000000 01111011	hexadecimal form
(unsigned short)65535	11111111 11111111	largest
(unsigned short)0	00000000 00000000	smallest

Type: double

Description: A (positive or negative) double-precision floating point number.

Size: System dependent. On FC010 with gcc217: 8 bytes.

Example Variable Declaration:

```
double dFirst;
```

Example Literals (assuming size is 8 bytes):

C Literal	Note
123.456	fixed-point notation
1.23456E2	scientific notation
.0123456	fixed-point notation
1.234546E-2	scientific notation with negative exponent
-123.456	fixed-point notation
-1.23456E2	scientific notation with negative mantissa
-.0123456	fixed-point notation
-1.23456E-2	scientific notation with negative mantissa and negative exponent
1.797693E308	largest (approximate)
-1.797693E308	smallest (approximate)
2.225074E-308	closest to 0 (approximate)

Type: float

Description: A (positive or negative) single-precision floating point number.

Size: System dependent. sizeof(float) <= sizeof(double). On FC010 with gcc217: 4 bytes.

Example Variable Declaration:

```
float fFirst;
```

Example Literals (assuming size is 4 bytes):

C Literal	Note
123.456F	fixed-point notation

```
1.23456E2F            scientific notation

.0123456F             fixed-point notation
1.234546E-2F          scientific notation with negative exponent

-123.456F             fixed-point notation
-1.23456E2F           scientific notation with negative mantissa

-.0123456F            fixed-point notation
-1.23456E-2F          scientific notation with negative mantissa and negative exponent

3.402823E38F          largest (approximate)
-3.402823E38F         smallest (approximate)
1.175494E-38F         closest to 0 (approximate)
```

Type: long double

Description: A (positive or negative) extended-precision floating point number.

Size: System dependent. sizeof(long double) >= sizeof(double). On FC010 with gcc217: 16 bytes.

Example Variable Declaration:

```
long double ldFirst;
```

Example Literals (assuming size is 16 bytes):

```
C Literal             Note

123.456L              fixed-point notation
1.23456E2L            scientific notation

.0123456L             fixed-point notation
1.234546E-2L          scientific notation with negative exponent

-123.456L             fixed-point notation
-1.23456E2L           scientific notation with negative mantissa

-.0123456L            fixed-point notation
-1.23456E-2L          scientific notation with negative mantissa and negative exponent

1.18973E4932L         largest (approximate)
-1.189731E4932L       smallest (approximate)
3.3621E-4932L         closest to 0 (approximate)
```

Differences between C and Java:

Java only:
boolean, byte

C only:
unsigned char, unsigned short, unsigned int, unsigned long
long double

Java: Sizes of all types are **specified**
C: Sizes of all types except char are **system dependent**

Java: char comprises **2** bytes
C: char comprises **1** byte