Lecture 4. Variables

A *Variable* is a **named** storage location (memory location), that stores a **value** of a particular **data type**.

Declaration syntax: **data_type variable_name;**

where **data_type** represents the variable data type
variable_name is a variable name

Multiple variables of the same data type may be declared in the same statement where names are separate by commas.

Example: int number1, number2, sum;

Variables can be initiated in the declaration statement by using the **assignment**

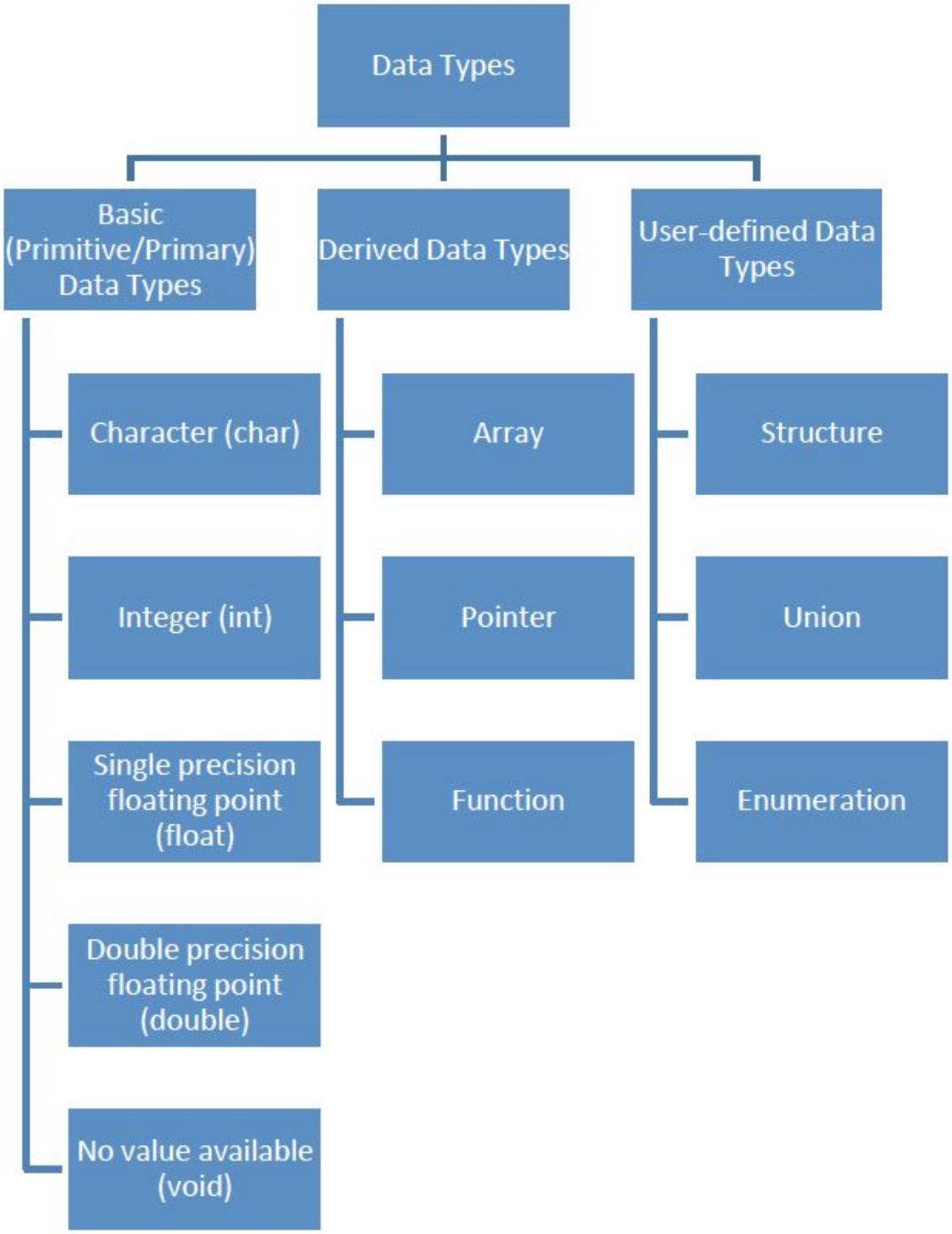
operator (**=**). Example: int number1 = 1;

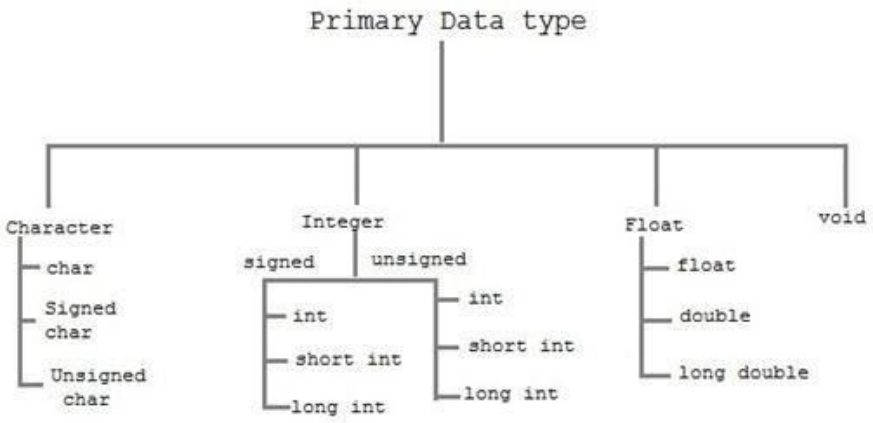
Variable name

Since the variable names are identifiers, the naming convention must follow the rules given for identifiers.

- Every variable must be declared before it is used.
- Variable can only store a value of a specified data type.
- A variable may take different values during the program execution.
- Each declaration statement must end with semi-colon (;).
- Variable names may contain, but not begin with, an integer.
- Variable names should be descriptive

Data types


Data Types
Basic (Primitive/Primary) Data Types
Derived Data Types
User-defined Data Types
Character (char)
Integer (int)
Single precision floating point (float)
Double precision floating point (double)
No value available (void)
Array
Pointer
Function
Structure
Union
Enumeration



"Primary Data Type." n.d. Online Image. Studytonight. 23 Jan, 2017. http://www.studytonight.com/c/datatype-in-c.php

Note: signed, unsigned, short, and long are **type modifiers**.

Data Type	Keyword	Storage size	Valuerange
Character	char	1 byte	-128 to 127 or 0 to 255
Unsigned character	unsigned char	1 byte	0 to 255
Signed character	signed char	1 byte	-128 to 127
Integer	int	2 OR 4 bytes	-32,768 to 32,767 OR -2,147,483,648 to 2,147,483,647
Unsigned integer	unsigned int	2 OR 4 bytes	0 to 65,535 OR 0 to 4,294,967,295
Short integer	short int	2 bytes	-32,768 to 32,767
Unsigned short integer	unsigned short int	2 bytes	0 to 65,535
Long integer	long int	4 bytes	-2,147,483,648 to 2,147,483,647
Unsigned long integer	unsigned long int	4 bytes	0 to 4,294,967,295
Float	float	4 bytes	-1.2E-38 to 3.4E+38 Precision: 6 decimal places
Double	double	8 bytes	2.3E-308 to 1.7E+308 Precision: 15 decimal places
Long double	long double	10 bytes	3.4E-4932 to 1.1E+4932 Precision: 19 decimal places

The sizes and ranges of different variable types are compiler dependent. To get the exact size of a data type, the operator **sizeof()** can be used. The expression **sizeof(data_type)** will return a number of bytes required to store a particular data type.

Constants (Literals)

A named memory location which holds a **fixed** value that cannot be modified by the program during its execution

Types of constants:

- Integer
  - Decimal – Example: 123
  - Octal – Using prefix 0. Example: 0123
  - Hexadecimal – Using prefix 0x or 0X. Example: 0x2A
- Floating point – Examples: 123.45, -0.2E-2
- Character
  - Examples: 'A', '1', '&'
  - Special Backslash character constants – Example: '\n'
- String – Example: "Seneca"

Defining Constants

There are two ways to define a constant in C:

- using **`#define`** preprocessor Examples:
  - `#define PI 3.14`
  - `#define NEWLINE '\n'`
- using **`const`** keyword Examples:
  - `const int SIZE = 100;`
  - `const float PI = 3.14;`
  - `const char NEWLINE = '\n';`

Example:

```
#include <stdio.h>
#define SIZE 10

int main()
{
   const float PI = 3.14;
   const char letter = 'A';
   printf("pi=%.2f\n", PI);
   printf("Section: %c\n", letter);
   printf("size = 2 x %d = %d\n", SIZE, 2*SIZE);
   return 0;
}
```

Output:

```
pi=3.14
Section: A
size = 2 x 10 = 20
```

Function scanf()

scanf() function reads data from the input device (usually keyboard) and store it in a variable. To use scanf() function, you will have to include header **<stdio.h>** (same as for printf()).

Syntax: **scanf("*format_string*", &*variable1, &variable2, …*);**

where

- *format_string* – Specifies the data type of each variable from the list. Common format specifiers:

Data Type	Format Specifier
int	%d
float	%f
double	%lf
char	%c
string	%s

- *Ampersand sign (&)* – "Address of" operator
  - Tells scanf() where (in memory) to store the new value entered by the user
  - Missing & in scanf() is a common error; it leads to abnormal program termination.

Example:

```
int number;
printf("Enter a number: ");     //User prompted to enter a number
scanf("%d", &number);           /*scanf reads an int value from the keyboard and stores
                                  it into variable number*/
```

After scanf() is called, the program waits for user to enter a value and press the "Enter" key.

scanf() can be used to enter multiple values, of different or same data types, as shown in the example below:

```
int age;
float
height;
printf("Enter your age and height: ");
scanf("%d%f", &age, &height);          // %d is used for variable age, %f for height
```

References

- Tan, H.H., and T.B. D'Orazio. *C Programming for Engineering & Computer Science*. USA: WCB McGRaw-Hill. 1999. Print.
- Hock-Chuan, Chua. *C programming Tutorial*. Programming notes, n.d. Web. 23 Jan, 2017. <https://www3.ntu.edu.sg/home/ehchua/programming/cpp/c1_Basics.html#zz-3.1>
- Tutorialspoint.com. "C Data Types." *Www.tutorialspoint.com*. N.p., n.d. Web. 02 Mar. 2017. <https://www.tutorialspoint.com/cprogramming/c_data_types.htm>.
- Tutorialspoint.com. "C Constants and Literals." *Www.tutorialspoint.com*. N.p., n.d. Web. 02 Mar. 2017. <https://www.tutorialspoint.com/cprogramming/c_constants.htm>.