KATHMANDU UNIVERSITY SCHOOL OF MANAGEMENT

BBIS

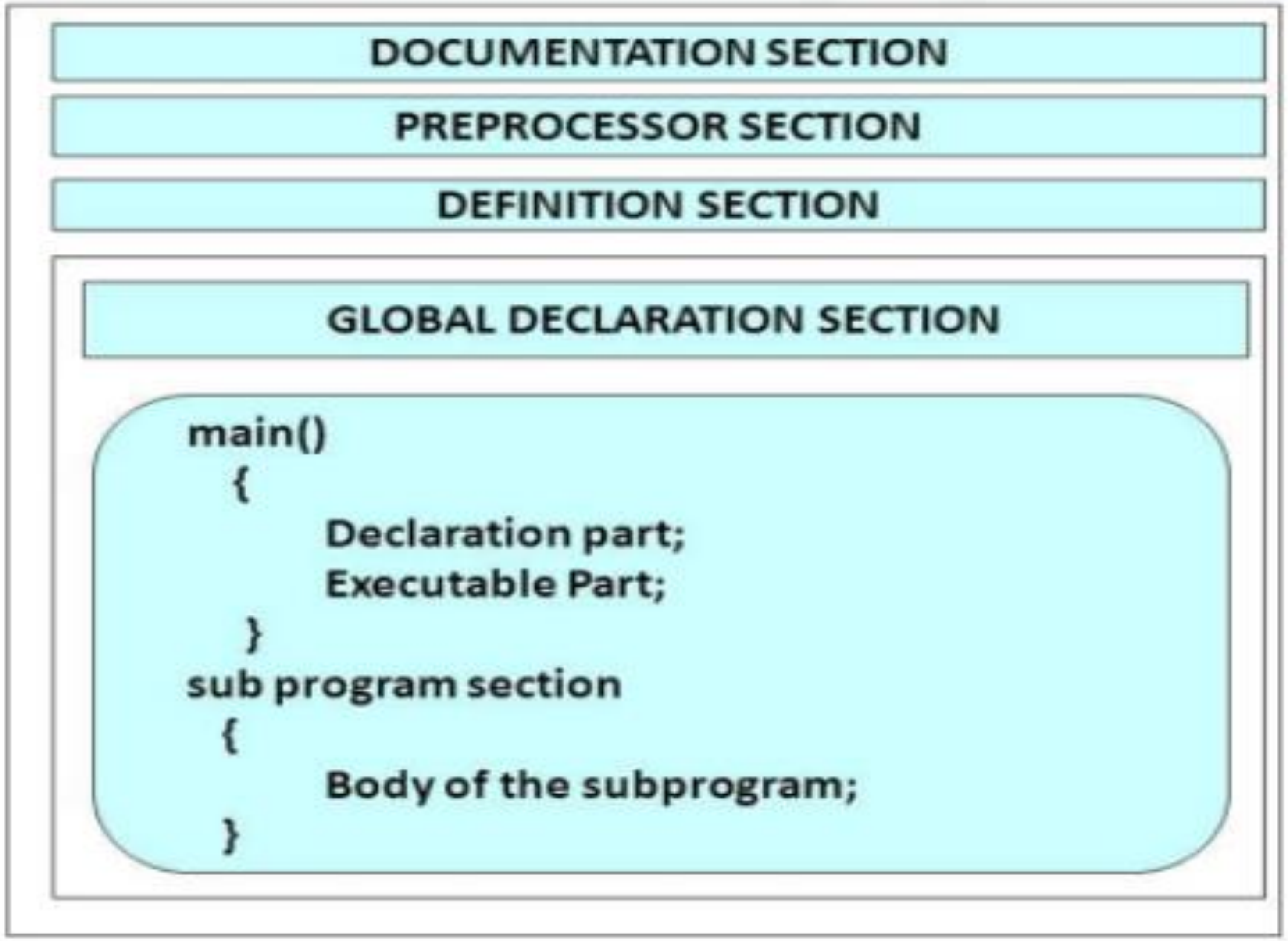
COM 102 : 3 Credit Hours

3. Fundamentals of C

02/01/2022

Outlines

Structure of C program



DOCUMENTATION SECTION

PREPROCESSOR SECTION

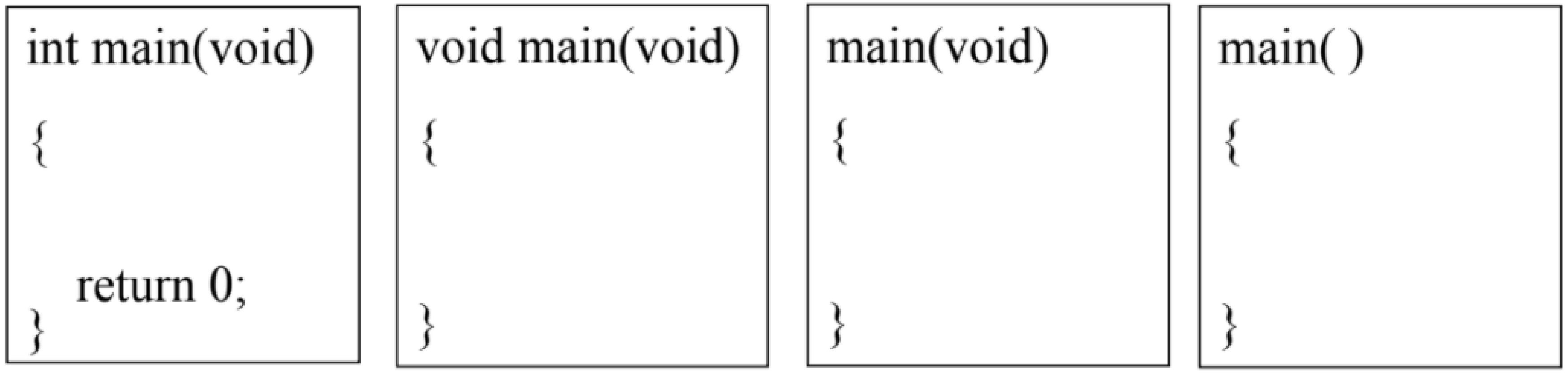
DEFINITION SECTION

GLOBAL DECLARATION SECTION

```
main()
  {
        Declaration part;
        Executable Part;
  }
sub program section
  {
        Body of the subprogram;
  }
```

Main function

- Identify the start of the program.
- Every C program has a main ( )
- 'main' is a C keyword.
    - not use it for any other variable name or function name.
-



```
int main(void)
{

   return 0;
}
```

```
void main(void)
{

}
```

```
main(void)
{

}
```

```
main( )
{

}
```

Statement

- C programs are collection of Statements,
  - statements is an executable part of the program it will do some action.
- Each statement in C needs to be terminated with semicolon (;)
- Example:

```
1   x = 2;              /* an assignment statement */
2   x = 2+3;            /* another assignment statement */
3   2+3;                /* has no effect---will be discarded by smart compilers */
4   puts("hi");         /* a statement containing a function call */
5   root2 = sqrt(2);    /* an assignment statement with a function call */
```

Expression Statements

Few Examples for expression Statements

- X = Y + 10 ;
- 20 > 90;
- a ? b : c ;
- a = 10 + 20 * 30;
- ; (This is NULL Statement).

Selection Statements

- are used in decisions making situations we will look about selections statements in Later Tutorials.

Examples:

- if
- if…else
- switch

Compound Statement

Enclosed within the Braces { }.
Compound statement is also called as Block Statement.

```
{
      int a=10,b=20,c;
      c = a + b;
      printf("value of C is : %d n",c);
}
```

Iterative Statements

also Called as Loops. If we want to Execute a part of program many times we will use loops.

Here is the List of Basic loops in C language.

- for loop.
- while loop.
- do-while loop.

Character Set in C Language

- The C Language have basic character set which includes the Alphabets, Digits, Special Characters, and Escape Sequences.
- C language supports a total of 256 characters.

Alphabets

- C-Programming language support all the 52 upper and lower case characters of Alphabets.
  - A,B,C,D.............Z
  - a,b,c,d,................z

Character Set in C Language

- Digits
  - C language supports 10 digits which are used to construct numerical values in C language.
  - Digits – 0, 1, 2, 3, 4, 5, 6, 7, 9

- Special Characters

The C Programming Language supports 29 special characters like brackets (Curly brackets, Square brackets), Quotes, Hash, Question mark and so on. Special Symbols - ~ @ # $ % ^ & * ( ) _ - + = { } [ ] ; : ' " / ? . > , < \ etc.,

- C Language also supports five White Space characters.
  - Spaces
  - Horizontal tab
  - Vertical tab
  - Newline
- Every character in C language has its equivalent ASCII (American Standard Code for Information Interchange) value.

Escape Sequences

- Escape sequences in programming is a special sequence of characters, which is used to escape the meaning of the sequence to give it a different meaning.
- All C Language characters are printed on console but some Characters such as new line character, tab, question mark(?), backslash(\ ), etc. can not be printed like normal characters.
- To print these characters we need to use Escape sequences.
- An escape sequence always starts with the back-slash (\ ) and is followed by one or more special characters.

List of Escape Characters in C

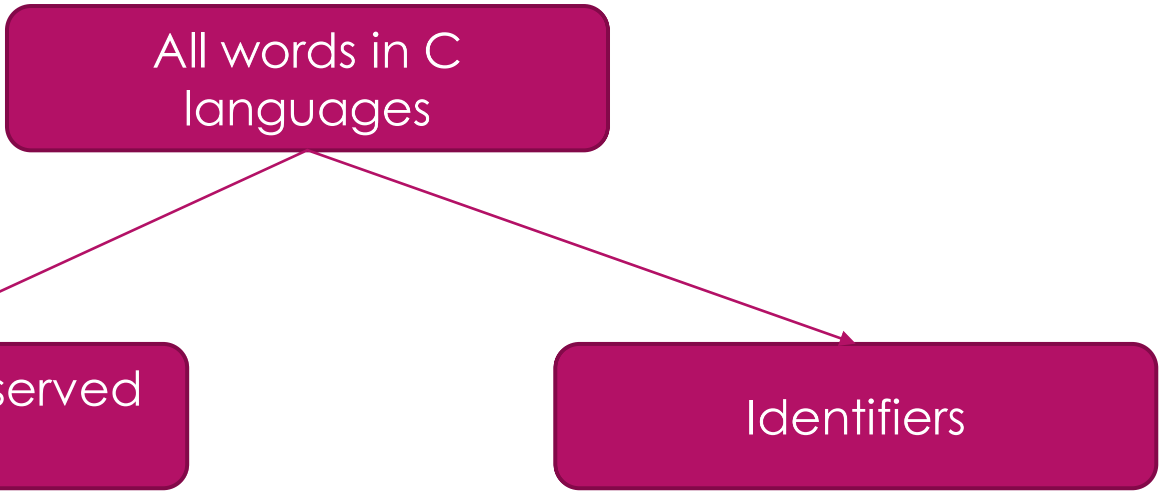
Character	Escape Sequence	ASCII Value
Back Space	`\b`	08
Bell	`\a`	07
Horizontal Tab	`\t`	09
New Line	`\n`	10
Vertical Tab	`\v`	11
Form Feed	`\f`	12
Carriage Return	`\r`	13
Quotation Mark	`\”`	34
Apostrophe	`\’`	039
Question Mark	`\?`	063
Backslash	`\\`	092
NULL		0

Keywords and Identifiers

In C Programming language we have two types of words one are keywords and others are **Identifiers**.

All the words we deal with in the C programming can be classified into two types. They are.

- Keywords or Reserved words
- Identifiers



Keywords in C

- There are some words in C programming like **if, while, int**, etc. which have a **predefined meaning for the C compiler**. These words are known as **Reserved words or Keywords.**
- **Keywords Examples :**
  - **int**, **float**, **double**,
- We have 32 Keywords in the C Programming Language.
  - we can't use these keywords for other purposes in a C Programming.
  - We can not use any keyword name for naming Variables, Functions, etc.

List of Keywords in C Programming

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

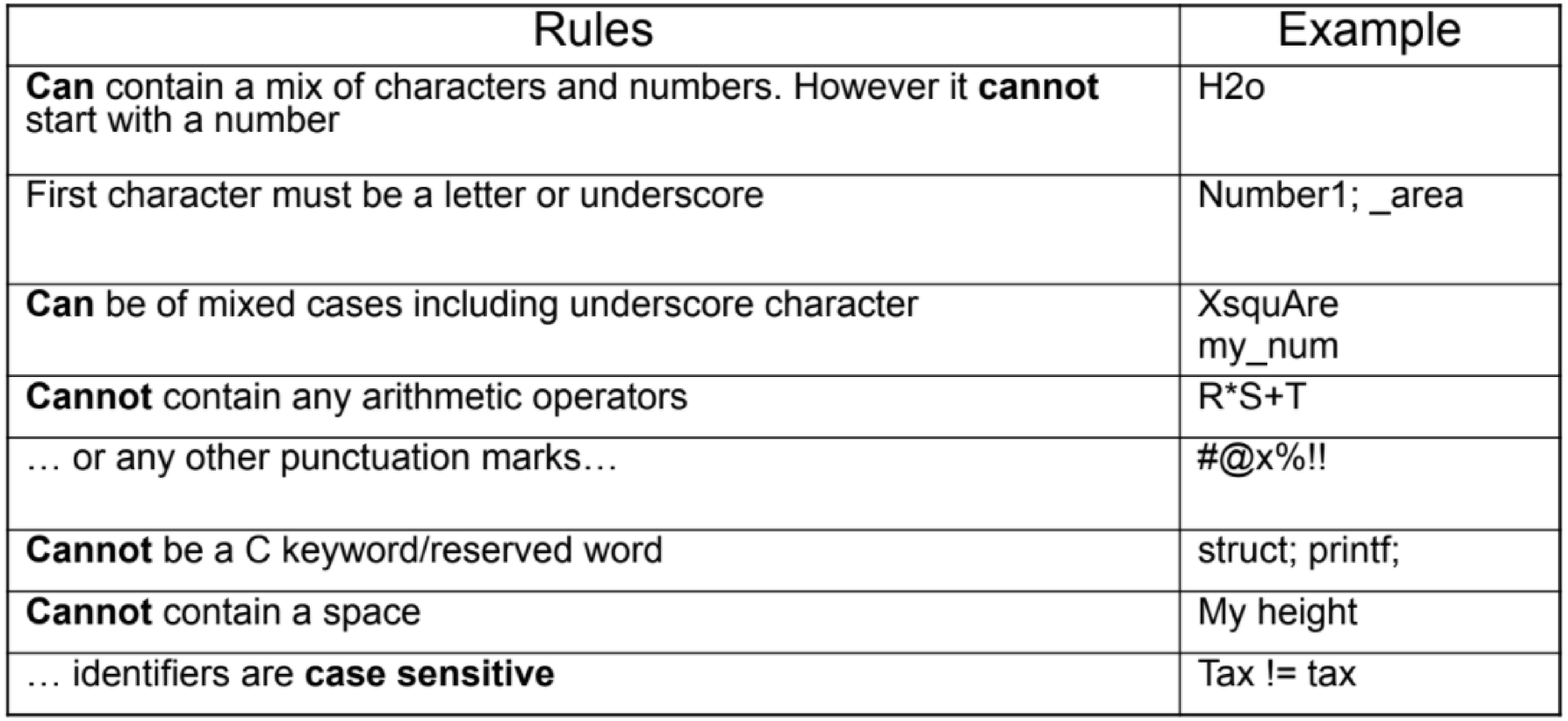
Identifiers in C

- The Identifiers in C Language are user defined words and unknown to the C Compiler.
- The Identifiers are used to name any entity like Functions, Variables, Arrays and Structures,..etc.
- to use Identifier then we have to define it's meaning.
- Example:
  - int my_name;
    - my_name is an identifier used as a program variable
  - void CalculateTotal(int value)
    - CalculateTotal is an identifier used as a function name

Rules for naming identifiers

- An identifier can only have alphanumeric characters (a-z , A-Z , 0-9) (i.e. letters & digits) and underscore(_) symbol.
- Identifier names must be unique.
- The first character must be an alphabet or underscore.
- You cannot use a keyword as identifiers.
- Only the first thirty-one (31) characters are significant.
- It must not contain white spaces.
- Identifiers are case-sensitive.

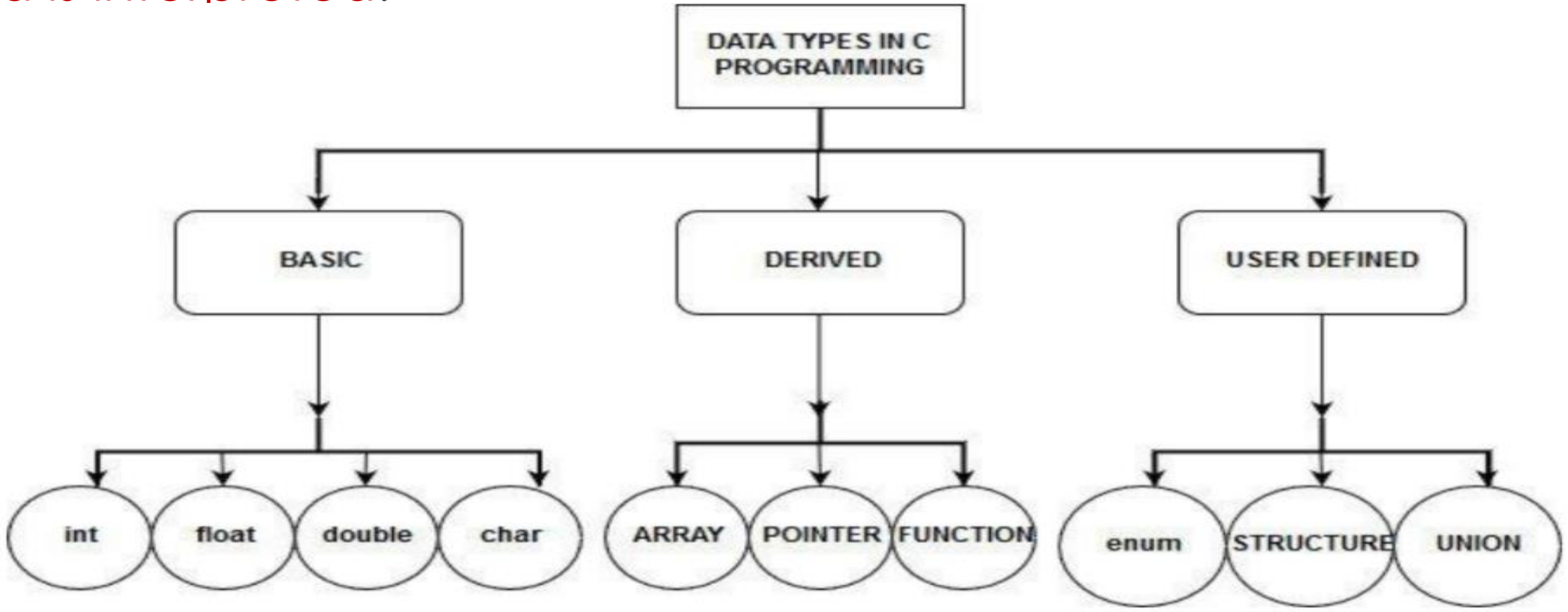
Rules for naming identifiers with



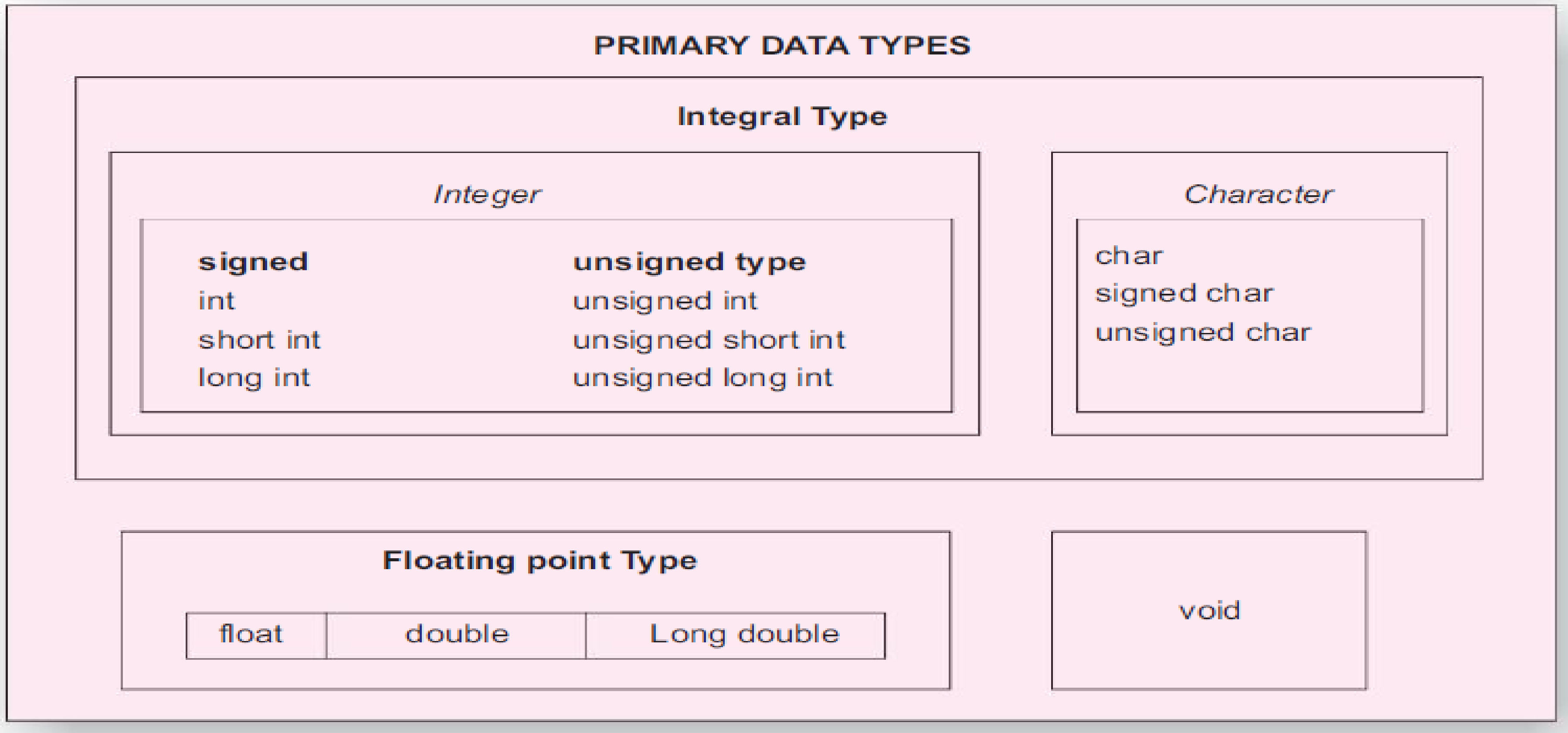
Rules	Example
Can contain a mix of characters and numbers. However it **cannot** start with a number	H2o
First character must be a letter or underscore	Number1; _area
Can be of mixed cases including underscore character	XsquAre my_num
Cannot contain any arithmetic operators	R*S+T
… or any other punctuation marks…	#@x%!!
Cannot be a C keyword/reserved word	struct; printf;
Cannot contain a space	My height
… identifiers are **case sensitive**	Tax != tax

3.3 Data Type, Variable Declaration

- Data types in c refer to an extensive system used for declaring variables or functions of different types.
- The type of a variable determines how much space it occupies in storage and how the bit pattern stored is interpreted.



Basic data types



Integer data types

- The Integer datatype is used to store the Integer data values.
- Examples of Integer data values are 10, 20, 100, etc.
  - int num = 10;

- The Integer datatype can further divided into two types.
  1. Signed Integer Data
     - **int a = -10;**
  2. Unsigned Integer Data
     - The Unsigned Integer datatypes only contains the Positive values. So Negative values are not allowed.
     - **int a = 10;**
- We use %d format specifier to represent the integer datatype values.

Float data types

- The Float Datatype is used for Storing Floating point values.

- Example of floating point data.

  10.56, 78.37, 10.0, etc

- Floating point data also called as the real data.
- The float keyword is used to create the floating point data variables in C Language.
- Example: **float pi = 3.14;**
- We use %f format specifier to represent the float datatype values.

Char data types

- Character Datatype is used to **Store Single Character**.
- Example of character data.
  'V' 'N', 's', etc

Only one character is allowed for character datatype.

- **char** ch = ‘V’

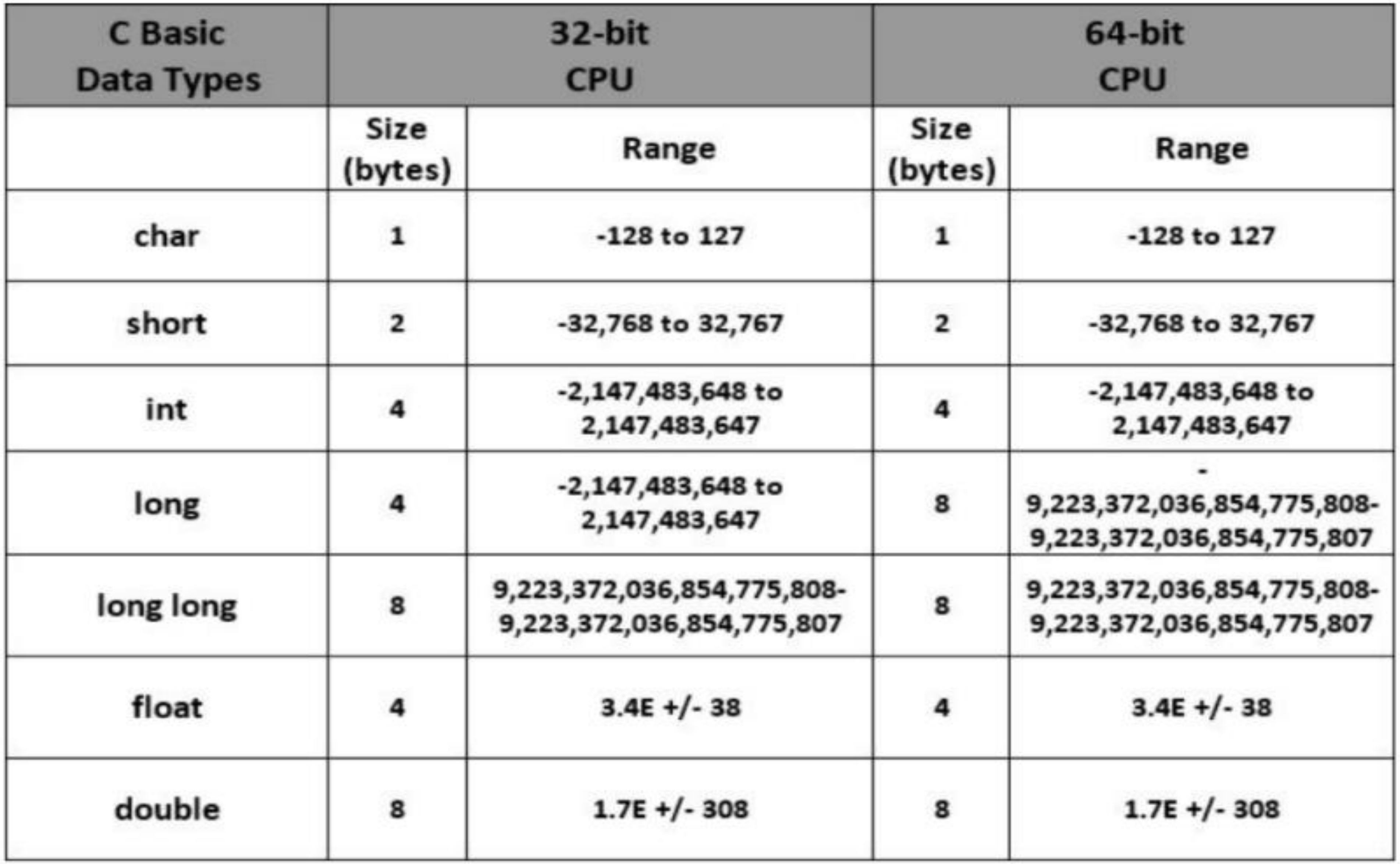
- We use %c format specifier to represent the character datatype values.

Void data types

- The void type has no values.
- The type of function is said to be void when it does not return any value to the calling function.

- The void keyword can also be used in some other contexts:
  - As the only parameter type in a function prototype, as in int func(void), it indicates that the function has no parameters. (C++ uses empty parentheses for this, but they mean something else in C.)
  - As the return type of a function, as in void func(int n), it indicates that the function returns no result.
  - void* is a pointer type that doesn't specify what it points to.

Data types and their size the can address



C Basic Data Types	32-bit CPU		64-bit CPU	
	Size (bytes)	Range	Size (bytes)	Range
char	1	-128 to 127	1	-128 to 127
short	2	-32,768 to 32,767	2	-32,768 to 32,767
int	4	-2,147,483,648 to 2,147,483,647	4	-2,147,483,648 to 2,147,483,647
long	4	-2,147,483,648 to 2,147,483,647	8	- 9,223,372,036,854,775,808- 9,223,372,036,854,775,807
long long	8	9,223,372,036,854,775,808- 9,223,372,036,854,775,807	8	9,223,372,036,854,775,808- 9,223,372,036,854,775,807
float	4	3.4E +/- 38	4	3.4E +/- 38
double	8	1.7E +/- 308	8	1.7E +/- 308

Variables

- Variable is a name associated with a memory cell whose value can change.
- Variables are used to store data value.
- These names are chosen by programmer.
- **Variable Declaration**: specifies the type of a variable
  - Example: int num;
- **Variable Definition**: assigning a value to the declared variable'
  - Example: num = 5;

Rules for writing variable names

- A variable name may consists of letters, digits and the underscore (_) characters.
- A variable name must begin with a letter. Some system allows to starts the variable name with an underscore as the first character.
- ANSI standard recognizes a length of 31 characters for a variable name. However, the length should not be normally more than any combination of eight alphabets, digits, and underscores.
- Uppercase and lowercase are significant. That is the variable Ram is not the same as ram and RAM.
- The variable name should not be a C reserved word (keyword).e.g switch can not be a valid variable name in c.

Naming Conventions for Variables

- Generally, C programmers maintain the following conventions for naming variables.
  - Start a variable name with lowercase letters.
  - Try to use meaningful identifiers
  - Separate "words" within identifiers with mixed upper and lowercase (for

    example: empCode) or underscores (for example emp_code).
  - For symbolic constants use all uppercase letters (for example #define LENGTH 100, #define MRP 45).

Constants

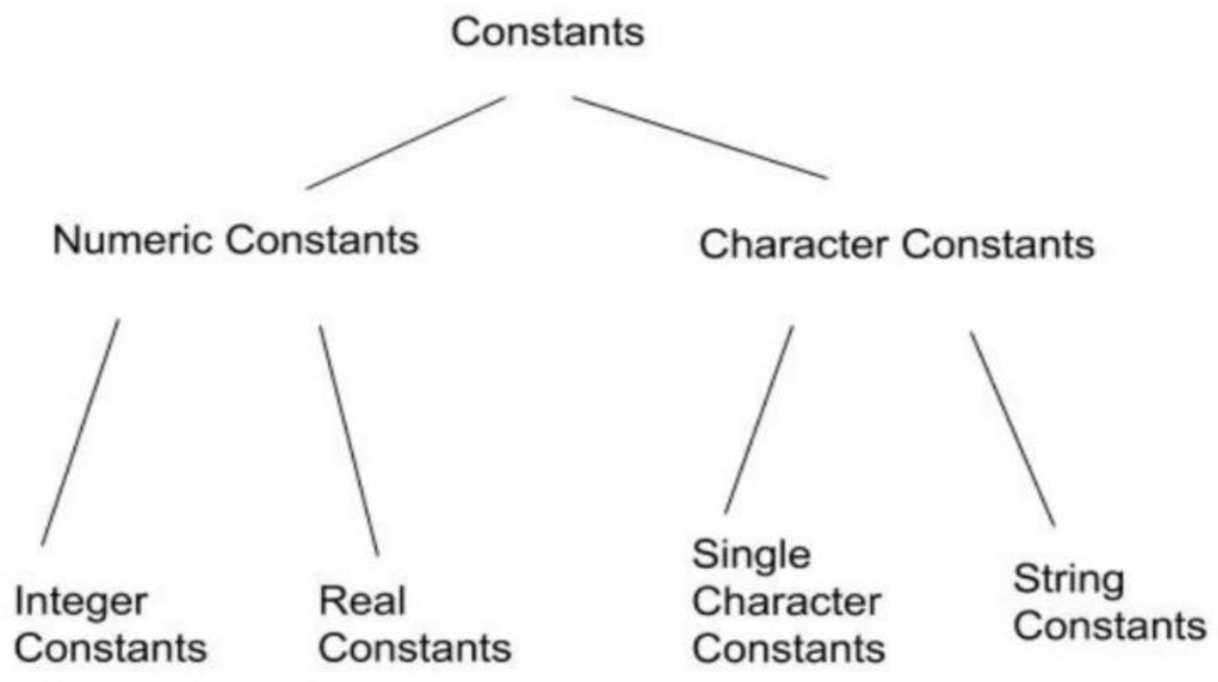
Constants in C are the fixed values that are used in a program, and its value remains the same during the entire execution of the program.

- Constants are also called literals.
- Entities that appear in the program code as fixed values.
- Any attempt to modify a CONSTANT will result in error.
- Constants can be any of the data types.
- We can use C constants in program with const keyword and #define preprocessor.

e.g.

- const char x = ‘A’;
- #define pi 3.412

Constants



Basic Types of C Constants

Types of Constants

Numeric Constant

- It contains constant of numeric values. For example, values like 1, 2, 1.3 1.4
- etc.

It is further divided into –

(1) Integer Constant. e.g. 10, 250, 1000

Decimal, Octal, Hexadecimal

(2) Real Constant. e.g. 0.042 , -0.452, 785.244, +452.00

…

- Character constant are the literal values. These are further divided into single character and string constants.
- Single Character Constants: Contains single character enclosed within a pair of single quote marks.
  - – const char letter = ‘n’;
  - – const char number = ‘1’;
  - – printf(“%c”, ‘letter’);
  - » Output would be: n

String Constants

- A character string, a string constant consists of a sequence of characters enclosed in double quotes.
- A string constant may consist of any combination of digits, letters, escaped sequences and spaces. Note that a character constant ‘A’ and the corresponding single character string constant "A" are not equivalent.
- Valid String Constants: -   "W"   "100"   "24, Newroad Street".
- Rules for Constructing String constants
  - A string constant may consist of any combination of digits, letters, escaped sequences and spaces enclosed in double quotes.
  - Every string constant ends up with a NULL character which is automatically assigned (before the closing double quotation mark) by the compiler.

Symbolic Constants

- A symbolic constant is a name given to any constant.
- In C, the preprocessor directive #define is used for defining symbolic constants.
- #define instructions are usually placed at the beginning of the program.
- By convention, the names of symbolic constants are written in uppercase, but this is not compulsory.

Creating constants in C

In a c programming language, constants can be created using two concepts...

1. **Using the 'const' keyword**
2. **Using '#define' preprocessor**

```
#include<stdio.h>
#include<conio.h>
void main(){
  int i = 9 ;
  const int x = 10 ;
  i = 15 ;
  x = 100 ; // creates an error
  printf("i = %d\nx = %d", i, x ) ;
}
```

```
#include<stdio.h>
#include<conio.h>
#defien  PI  3.14

void main(){
  int r, area ;

  printf("Please enter the radius of circle : ") ;
  scanf("%d", &r) ;

  area = PI * (r * r) ;

  printf("Area of the circle = %d", area) ;
}
```

Program to find the ascii value of C character sets

```
#include<stdio.h>
int main()
{
      printf("\n\n\t Computer Programming- COMP 102\n\n\n");
      char c;
      printf("Enter a character : ");
      scanf("%c" , &c);
      printf("\n\nASCII value of %c = %d",c,c);
      printf("\n\n\t KU - School of Engineering\n\n\n");
      return 0;
}
```

WAP to convert temperature from Celsius to Fahrenheit

- I. Input temperature in Celsius from user. Store it in some variable say celsius.
- II. Apply formula to convert the temperature to Fahrenheit i.e. fahrenheit = (celsius * 9 / 5) + 32.
- III. Print the value of fahrenheit.