CS F111: Computer Programming

(Second Semester 2020-21)

Lect 7: C Programming

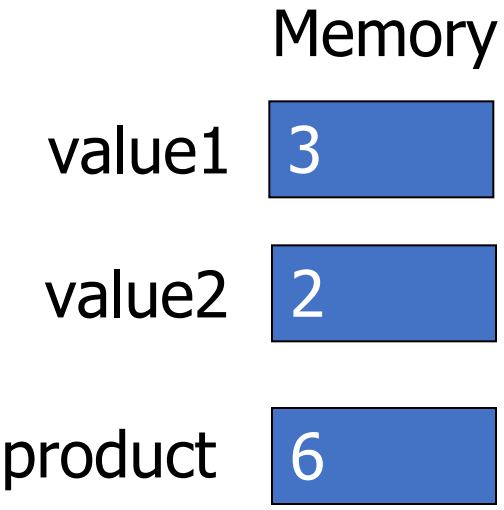
Dr. Nikumani Choudhury
Asst. Prof., Dept. of Computer Sc. & Information Systems
nikumani@hyderabad.bits-pilani.ac.in

BITS Pilani

Hyderabad Campus

Variable and Memory

```
#include <stdio.h>
int main( void )
{   int value1, value2, product ;
    printf(“Enter two integer values:”) ;
    scanf(“%d%d”, &value1, &value2) ;
    product = value1 * value2 ;
    printf(“Product = %d\n”, product) ;
    return 0;
    }
```



Every variable has a name, type, size, and value.

User defined Functions

```
main( )
{
    printf("\n I am in main") ;
    india();
    usa();
}
```

```
india( )
{
    printf("\n I am in india") ;
}
```

```
usa( )
{
    printf("\n I am in usa\n") ;
}
```

What is the output?

```
bash$ ./a.out
I am in main
I am in india
I am in usa
bash$
```

Notes to remember

- Any function can be called from any other function
- Order of function definitions need not be same as order of calls
- Two types: Library functions and User defined functions

Function declaration

```
main( )
{
    float a,b;
    printf(“\n Enter any number”);
    scanf (“%f”, &a);
    b=square (a);
    printf (“\n Square is %f”, b);
}
square ( float x )
{
    float y;
    y=x*x;
    return (y);
}
```

More in later chapters…

Good Programming Style

Rule #1: Use good (meaningful) names

```
int a;                  // BAD!!
int b;          // BAD!!

int radius;         // GOOD
int area;       //GOOD
```

Rule #2: Use indentation

```
int main(int argc, char *argv[])
{
    ...
    while (x == y) {
        something();
        somethingelse();

        if (some_error)
            do_correct();
        else
            continue_as_usual();
    }

    finalthing();
    ...
}
```

Continued…

Rule #3: Use whitespaces

```
a=3.14*radius*radius      // BAD!!
```

```
a = 3.14 * radius * radius;     //GOOD
```

Rule #4: Use blank lines to improve readability

```
int a = 10;
int b = 20;

if (a < b)
   printf (“ Ok”);

return 0;
```

Rule #5: Do not duplicate tests

```
if (basePay < 8.0) {
                ...
} else if (hours > 60) {
                ...
} else if (basePay >= 8.0 && hours <= 60) {
                ...
}
```

Continued…

Rule 6: Usage of parenthesis

```
while (a < b) {
…
}
```

```
function ( )
{
…
}
```

```
if (a < b) {
…
}
else {
…
}
```

Rule 7: Usage of semi-colon

```
if (a < b);
{
   …
}
```

Continued…

Rule 8: Commenting for clarity

1.// This is a line comment

2./* This is a block comment

3.over two lines */

Rule 9: Declarations should be provided at Col 1

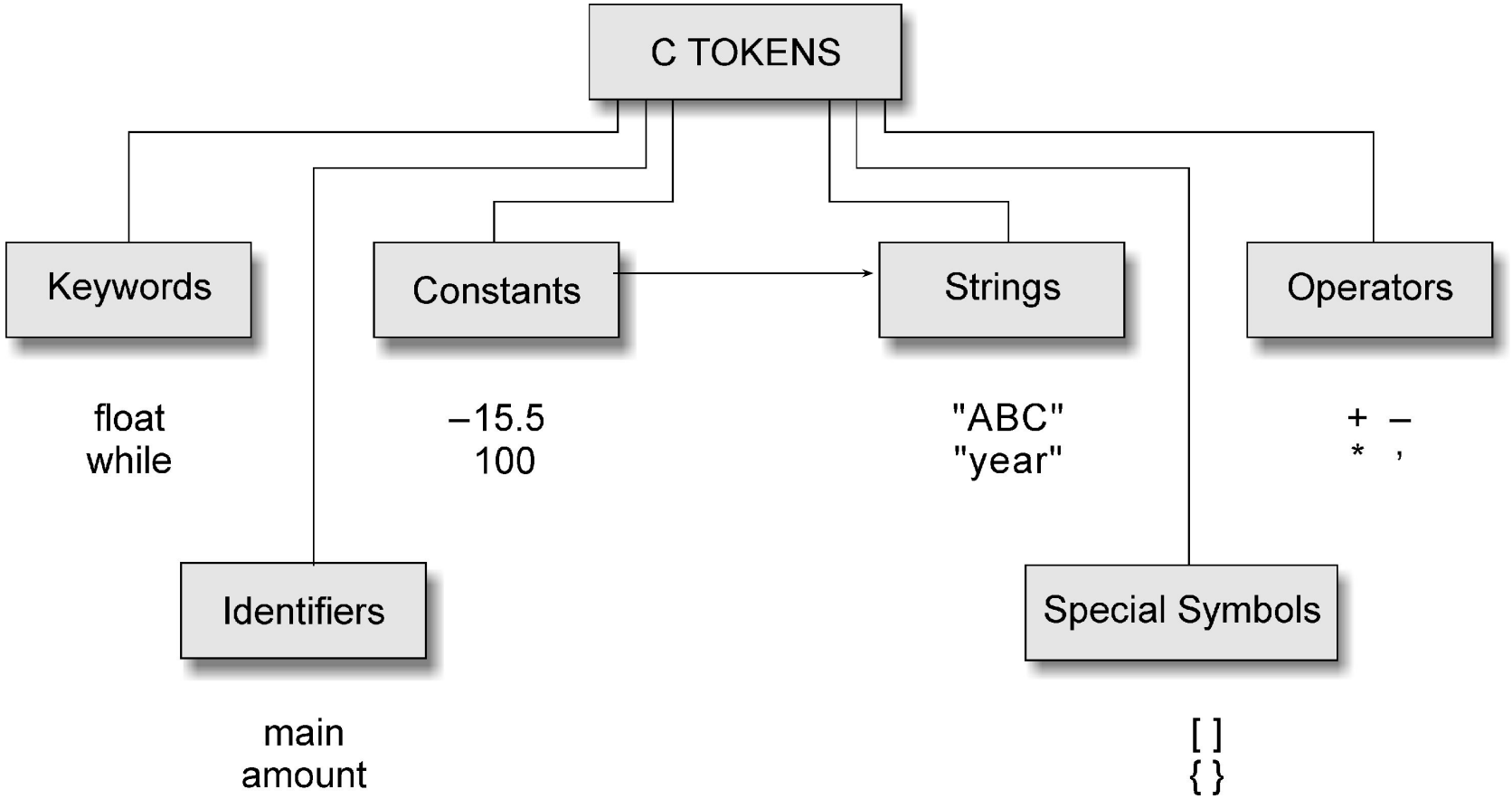
```
int x = 1;
char *msg = "message";
```

The C Character Set

- **Alphabets:** (52)
  - A, B, C, …Z and a, b, c, d, …z
- **Digits:** (10)
  - 0, 1, 2, …9
- **Special Symbols:** (31)
  - ~ ‘ ! @ # % ^ & * ( ) _ - + = | \ { } [ ] : ; “ ’ < > , . ? /
- **White spaces:**
  - Blank ( )
  - Tab (\t)
  - CR (\r) – moves the cursor to the beginning of the current line
  - New line (\n)

C Tokens

- **Smallest individual units**
- **Also known as lexical units**



Keywords and Identifiers

- Keywords are reserved words that have predefined meaning
- lowercase

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

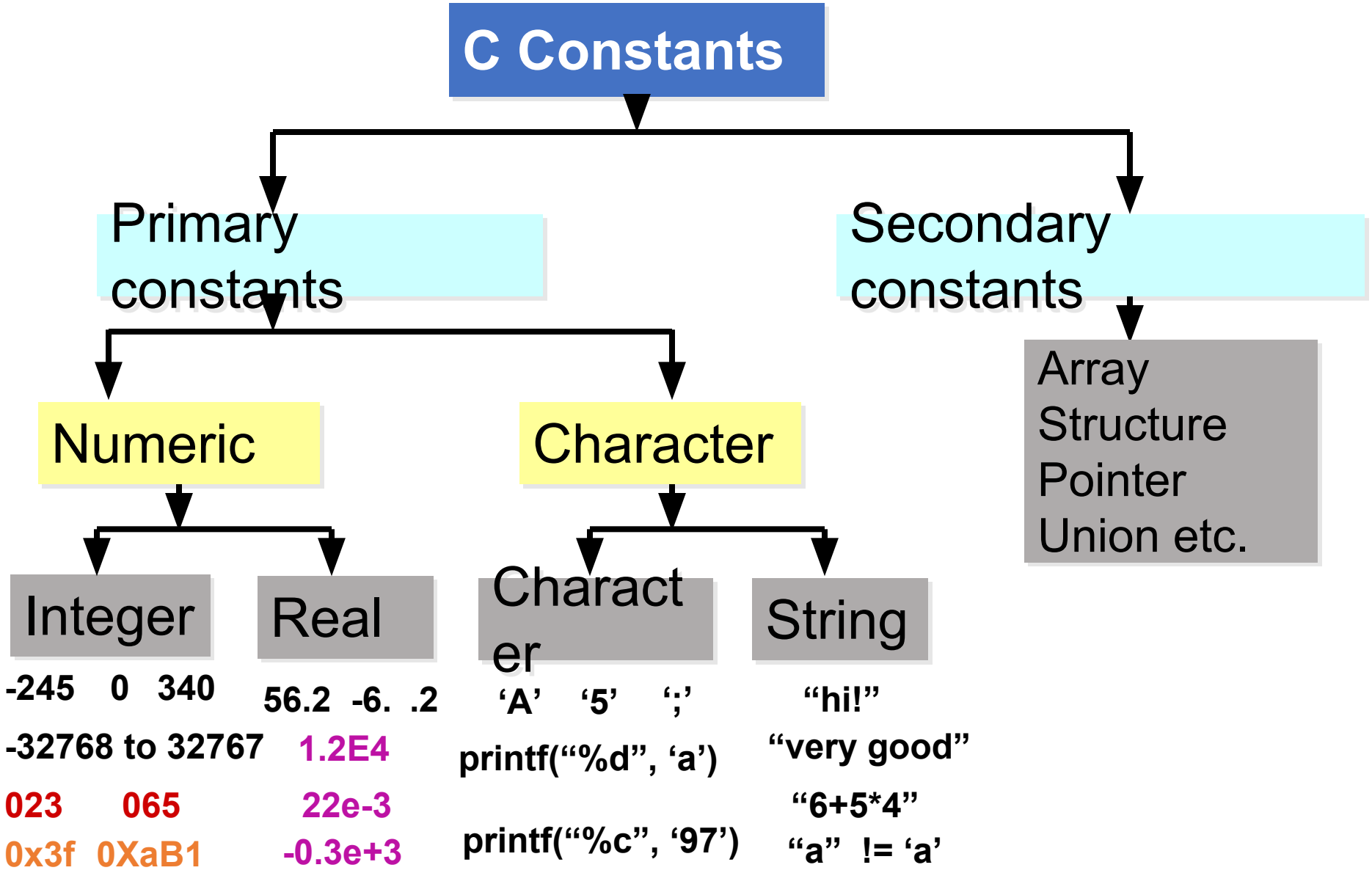
- **Identifiers: Names given to variables, functions and arrays**
  - **Case sensitive; Consists of letters, digits and underscore**

Rules for Naming Identifier

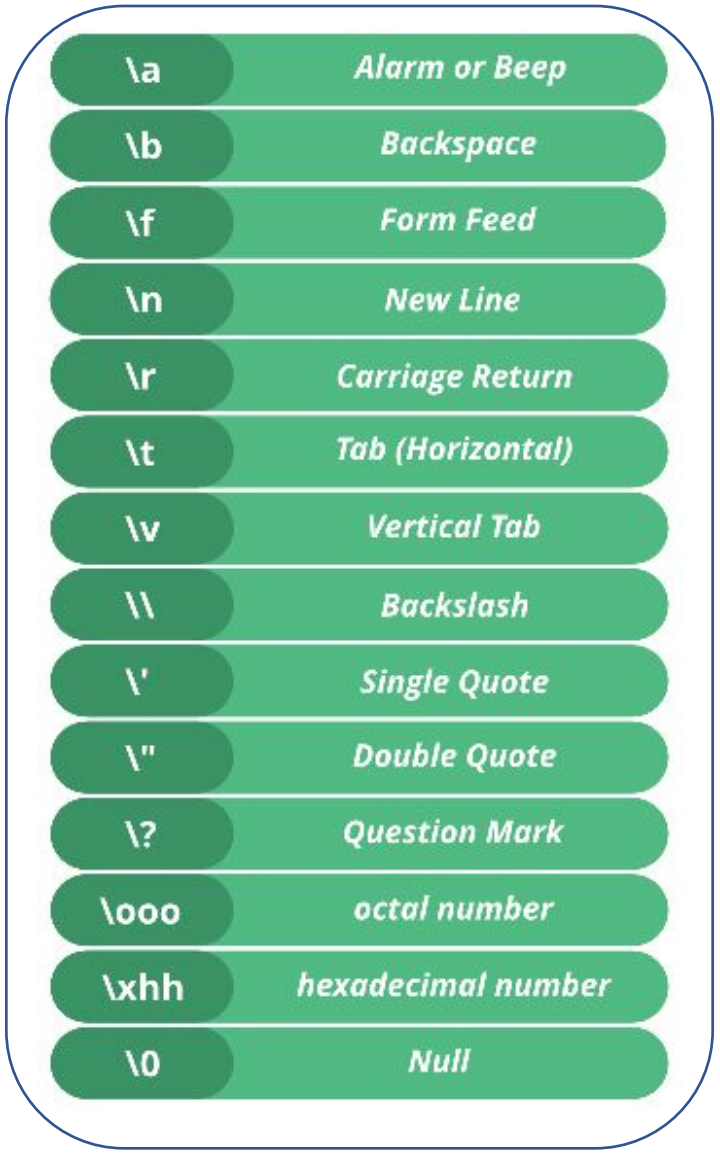
1. **An identifier must consist of only letters, digits or underscores.**
2. ***First character must be a letter or underscore***
3. **An identifier can be of arbitrary length.**
   - **Some C compilers recognize only the first few characters of the name (16 or 31).**
4. **Cannot be a reserved word i.e. keyword**
5. **Case sensitive**
   - area, AREA and Area are all different.

simple_interest ~~3_sum~~ count Average_1 ~~"hello"~~ _flag ~~(product)~~

Constants


C Constants
Primary constants
Secondary constants
Numeric
Character
Array
Structure
Pointer
Union etc.
Integer
Real
Character
String
-245 0 340
56.2 -6. .2
'A' '5' ';'
"hi!"
-32768 to 32767
1.2E4
printf("%d", 'a')
"very good"
023 065
22e-3
"6+5*4"
0x3f 0XaB1
-0.3e+3
printf("%c", '97')
"a" != 'a'

Constants Continued…



Escape sequences

Are these two same?

‘a’

“a”