SNS COLLEGE OF ENGINEERING

Kurumbapalayam (Po), Coimbatore – 641 107

An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

COURSE NAME : 23ITT101- PROBLEM SOLVING AND C PROGRAMMING

I YEAR /I SEMESTER

Unit 2- C-Programming Basics

Topic 4: Data types and Storage Classes

Brain Storming

1. What is Data types?
2. What is storage classes?

Example C Program

```
#include<stdio.h>
void main( )
{
int a, b, sum;
printf("\nEnter two no: ");
scanf("%d %d", &a, &b);
sum = a + b;
printf("Sum : %d", sum);
}
```

OUTPUT:

Enter two no:5 6

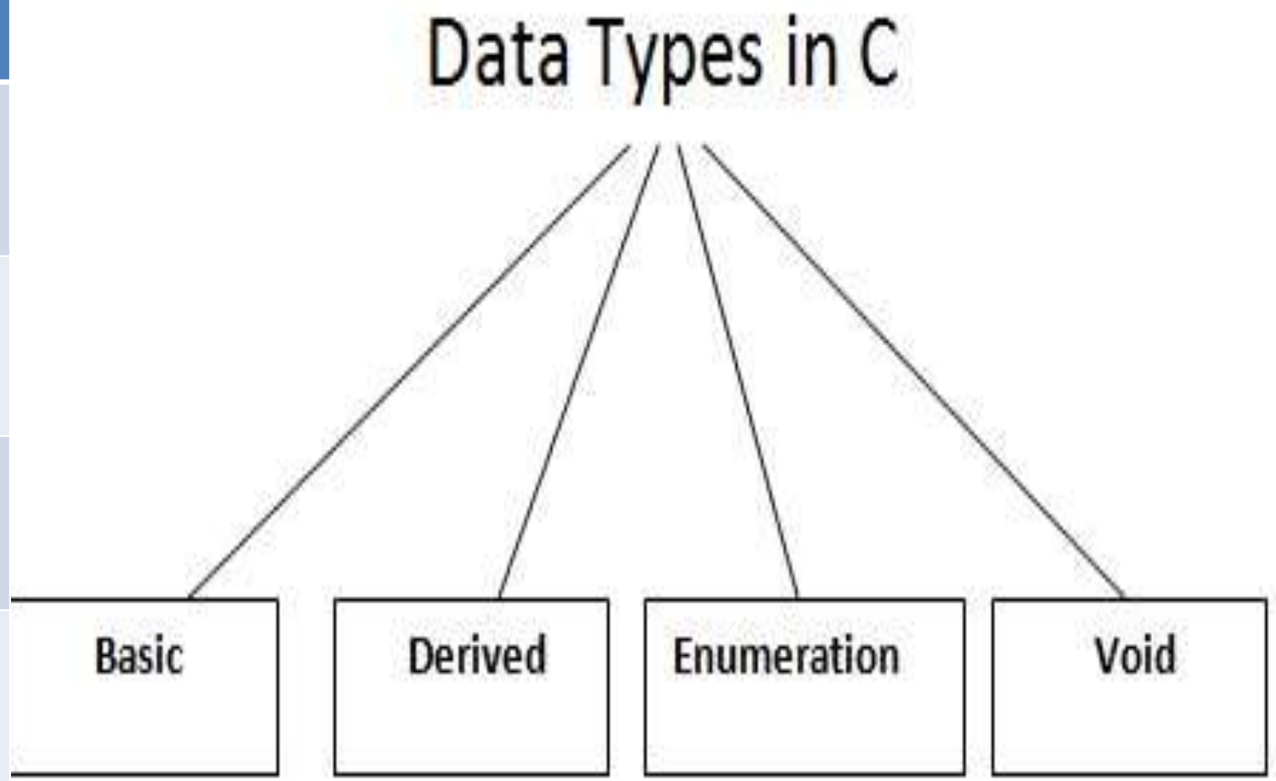
Sum:11

Data Types in C

- A data type specifies the type of data that a variable can store such as integer, floating, character, etc.
- 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.

Data Types in C

Types	Data Types
Basic Data Type	int, char, float, double
Derived Data Type	array, pointer, structure, union
Enumeration Data Type	enum
Void Data Type	void



Basic Data Types

- The memory size of the basic data types may change according to 32 or 64-bit operating system.
- Let's see the basic data types. Its size is given **according to 32-bit architecture**.

Datatypes and its Range

Data Types	Memory Size	Range
char	1 byte	−128 to 127
signed char	1 byte	−128 to 127
unsigned char	1 byte	0 to 255
short	2 byte	−32,768 to 32,767
signed short	2 byte	−32,768 to 32,767
unsigned short	2 byte	0 to 65,535
int	2 byte	−32,768 to 32,767
signed int	2 byte	−32,768 to 32,767
unsigned int	2 byte	0 to 65,535

Conti...

Data Types	Memory Size	Range
short int	2 byte	−32,768 to 32,767
signed short int	2 byte	−32,768 to 32,767
unsigned short int	2 byte	0 to 65,535
long int	4 byte	-2,147,483,648 to 2,147,483,647
signed long int	4 byte	-2,147,483,648 to 2,147,483,647
unsigned long int	4 byte	0 to 4,294,967,295
float	4 byte	
double	8 byte	
long double	10 byte	

Format Specifier

Format Specifier	Type
%c	Character
%d	Signed integer
%e or %E	Scientific notation of floats
%f	Float values
%g or %G	Similar as %e or %E
%hi	Signed integer (short)
%hu	Unsigned Integer (short)
%i	Unsigned integer
%l or %ld or %li	Long
%lf	Double
%Lf	Long double

%lu	Unsigned int or unsigned long
%lli or %lld	Long long
%llu	Unsigned long long
%o	Octal representation
%p	Pointer
%s	String
%u	Unsigned int
%x or %X	Hexadecimal representation
%n	Prints nothing
%%	Prints % character

Keywords

A keyword is a **reserved word**. You cannot use it as a variable name, constant name, etc. There are only 32 reserved words (keywords) in the C language.

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

Scope/Storage class of variables

- A scope in any programming is a region of the program where a defined variable can have its existence and beyond that variable it cannot be accessed.
- A storage class defines the scope (visibility) and life-time of variables and/or functions within a C Program.
- These features basically help us to trace the existence of a particular variable during the runtime of a program.
- We have four different storage classes in a C program.

Conti...

Storage classes in C

Storage Specifier	Storage	Initial value	Scope	Life
auto	stack	Garbage	Within block	End of block
extern	Data segment	Zero	global Multiple files	Till end of program
static	Data segment	Zero	Within block	Till end of program
register	CPU Register	Garbage	Within block	End of block

The auto Storage Class

- The **auto** storage class is the default storage class for all local variables.
- This is the default storage class for all the variables declared inside a function or a block.
- Hence, the keyword auto is rarely used while writing programs in C language.
- Auto variables can be only accessed within the block/function they have been declared and not outside them (which defines their scope).

Conti…

- Of course, these can be accessed within nested blocks within the parent block/function in which the auto variable was declared.
- However, they can be accessed outside their scope as well using the concept of pointers.

Example

```
#include <stdio.h>
int main( )
{
        auto int i = 1;
        {
                auto int i = 2;
                {
                        auto int i = 3;
                        printf ( "\n%d ", i);
                }
                printf ( "%d ", i);
        }
        printf( "%d\n", i);
}
```

OUTPUT:

3
2
1

The extern Storage Class

- The **extern** storage class is used to give a reference of a global variable that is visible to ALL the program files.
- When you use 'extern', the variable cannot be initialized however, it points the variable name at a storage location that has been previously defined.

First File: main.c

```
#include <stdio.h>
int count ;
extern void write_extern();
main()
{
count = 5;
write_extern();
}
```

Second File: support.c

```
#include <stdio.h>
extern int count;
void write_extern(void)
{
printf("count is %d\n", count);
}
```

Result:

count is 5

The register Storage Class

- The **register** storage class is used to define local variables that should be stored in a register instead of RAM.
- This storage class declares register variables which have the same functionality as that of the auto variables.
- This makes the use of register variables to be much faster than that of the variables stored in the memory during the runtime of the program.
- An important and interesting point to be noted here is that we cannot obtain the address of a register variable using pointers.

Example

```
#include <stdio.h>

int main()

{

register int i = 10;

int *p = &i; //error: address of register

variable //requested

printf("Value of i: %d", *p);

printf("Address of i: %u", p);

}
```

The static Storage Class

- Static variables have a property of preserving their value even after they are out of their scope.
- Hence, static variables preserve the value of their last use in their scope.
- So we can say that they are initialized only once and exist till the termination of the program.

Conti...

- Their scope is local to the function to which they were defined.
- Global static variables can be accessed anywhere in the program.

Example C Program

```c
#include<stdio.h>

void test();    //Function declaration (discussed in next topic)

int main()
{
    test();
    test();
    test();
}

void test()
{
    static int a = 0;       //a static variable
    a = a + 1;
    printf("%d\t",a);
}
```

OUTPUT:

```
1 2 3
```

Assessment 1

1. What is C Program?

Ans : _____

2. What are all the Features of C Program?

Ans : _____

References

TEXT BOOKS

1.E.Balagurusamy, “Fundamentals of Computing and Computer Programming”, 2nd Edition Tata McGRaw-Hill Publishing Company Limited, (2012). (UNIT – I, II, III, IV, V)

2.Ashok.N.Kamthane,“ Computer Programming”, Pearson Education (India) (2010). (UNIT –II, III IV, V)

3.Reema Thareja, “Programming in C”, 2nd Edition, Oxford University Press,(2015). (UNIT –I,II, III, IV, V)

REFERENCES

1.Byron Gottfried, “Programming with C”, 2nd Edition, (Indian Adapted Edition), TMH Publications, (2006). (Unit II, III, IV)

2.Stephan G kochan, “Programming in C” Pearson Education (2008), (UNIT II, III, IV, V)

3.P.Sudharson, “Computer Programming”, RBA Publications (2008), (UNIT I, II, III, IV)

4.Yashavant P. Kanetkar. “Let Us C”, BPB Publications, 2014.(Unit II, III, IV, V)

5.Anita Goel and Ajay Mittal, “Computer Fundamentals and Programming in C”, Dorling Kindersley (India) Pvt. Ltd., Pearson Education in South Asia, 2011. (UNIT – I, II, III, IV, V)

Thank You