1. **What is variable in C language? How to**

**define it with suitable example?**

1. It is memory location where we can store value such as integer, float, character.
2. Every variable must have memory address.
3. Depending upon data type of variable it takes memory.
4. Memory for the variable created generally on RAM and CPU register (when variable define as register).
5. Variable have different storage classes such as auto, static, register, and extern.
6. **There are two way to define variable.**
7. Declaration of variable.
8. Initialization of variable.
9. **Declaration of variable:-**
10. In that we simply define variable without assigning the value.
11. Example:-

int a;

1. Here a is variable of type integer and there is no value assign to variable a.
2. Hence it is called as declaration of variable.
3. **Initialization of variable:-**
4. In that we define variable with assigning value.
5. Example:-

int a=10;

1. Here a is variable of type integer and there is 10 value assign to variable a.
2. Hence it is called as initializationof variable.
3. Syntax for declaration of variable is

datatype variablename1,variablename2,………..variablenamen;

Example:-

int a,b,c;

1. Syntax for declaration of variable is.

datatype variablename1=value1,..variablenamen=valuen;

Example:-

int a=10,b=20,c=30;

1. Program for declaration of variable.

#include<stdio.h>

#include<conio.h>

void main()

{

int a;

clrscr();

getch();

}

1. Program for initializationof variable.

#include<stdio.h>

#include<conio.h>

void main()

{

int a;

clrscr();

getch();

}

1. **What are the types of variable in C**

**language along with example?**

1. There are three types of variable.
2. Local variable.
3. Global variable.
4. Environment variable.
5. Tree diagram for types of variable.

Types of variable

Environment variable

Global variable

Local variable

**Fig:-Types of Variable**

1. Explaination for above types of variable are given below.
2. **Local variable:-**
3. A variable define within body of any function is called as **Local variable**.
4. The scope of the local variable is visible within the body of function where it is define.
5. Also life time of local variable is in function where it is define.
6. We can’t access local variable outside the body of function.
7. By default storage class for local variable is **auto**.
8. But we can change default storage class of local variable into static and register.
9. Example:-

#include<stdio.h>

#include<conio.h>

void main()

{

int a=10;

clrscr();

printf(“Value of a which is local is=%d”,a);

getch();

}

1. Here a is local variable because a is define in main function.
2. **Note:-** Please do not think that variable define in main function is local variable. Local variable define within main function or it define in user define function.
3. **Global variable:-**
4. A variable define outside body any function is called as **Global variable**.
5. The scope of the global variable is visible anywhere in program.
6. Also life time of global variable is thought the program.
7. We can access global variable in the body of any function.
8. By default storage class of the global variable is **static**.
9. We can’t change storage class of global variable into auto and register. If you are trying to do this we will get compile time error. See example 9 below.
10. Default value for global variable is 0.
11. Example

#include<stdio.h>

#include<conio.h>

int a=10;

void main()

{

clrscr();

printf(“Value of a which is global is=%d”,a);

getch();

}

Here a is define as global variable.

1. Any change applied to global variable in the body of function is applicable to all another function in that program.

**Example:-**

#include<stdio.h>

#include<conio.h>

void myFunc();

int a;

void main()

{

a=10;

clrscr();

printf(“Value of a which is global is from main

function=%d”,a);

myFunc();

getch();

}

void myFunc()

{

printf(“Value of a which is global is from myFunc

function=%d”,a);

}

1. Example for point no. 6

#include<stdio.h>

#include<conio.h>

auto int a=10;

void main()

{

clrscr();

printf(“Value of a which is global is=%d”,a);

getch();

}

Auto storage class is not allowed here error was

generated.

1. **Environmental variable:-**
2. This variable available for C language program.
3. We do not need to initialize or declaring this variable.
4. This variable is accessible from anywhere in program. That means environmental variable has scope within program.
5. There are three functions are used for modifying and assigning environment variable. Those functions are given below.
6. setenv().
7. getenv().
8. putenv().
9. These function prototype is present in #include<stdlib.h> header file. So that while using these function includes #include<stdlib.h> header file in program.
10. Example:-

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

void main()

{

clrscr();

printf(“%s”,getenv(“DIR”));

printf(“\n%s”,getenv(“PATH”));

getch();

}

getenv function return null if the environmental variable set

or exits.

1. **What are rules to declared and initialized**

**the variable in C language along with**

**examples?**

**OR**

**What are naming rules for variables in C**

**Language?**

* Rules for declaring and initiation of variables in C language are as

follow.

1. Variable must start with alphabet or underscore.
2. Variable name must not have white space.
3. Keywords are not used as variable name.
4. Special symbols are used as variable name expect underscore character.
5. We can use digit in variable name but digit must not place starting of variable name.
6. Example for valid (legal) variables name.
7. int number;
8. float \_number;
9. float \_number\_;
10. int number1;
11. int \_\_;
12. float \_\_;
13. int num\_ber;
14. Example for invalid (illegal) variables name.
15. int switch;
16. float number$;
17. int 1number;
18. int num ber;
19. float $number;
20. int num@ber;
21. **What are standard rules to define**

**variable according software**

**engineering principle?**

**OR**

**What are naming convention for**

**variable?**

1. Variable should have meaningful name.
2. Variable name should start with small letter.
3. If two name are joining with each other. Then first letter of the second variable should be capital letter.
4. If the variable is constant then all letters of variables is capital and different name are joining with each other by underscore.
5. Example:-
6. int number;
7. int myVariable;
8. const int FIRST\_NUMBER=10;
9. **Note:-**
10. **Above naming convention for variable is recommendation.**
11. **Either we follow it or ignore it.**
12. **C language neither force to follow it nor give error if do not follow it.**
13. **But I personally suggest that to follow to become better developer and tester.**

**Point for your knowledge:-**

1. Memory for local variable is created in **stack memory**.
2. Variable allocated memory in two ways which are
3. Static memory allocation.
4. Dynamic memory allocation.
5. **Static memory allocation:-**
6. Memory for variable is created at compile time is called as **Static memory allocation**.
7. It is suitable for fixed size of variable.
8. Example:-

int a,b;

Here memory for a and b variables is allocated statically.

1. **Dynamic memory allocation:-**
2. Memory for variable is created at run time is called as **Dynamic memory allocation**.
3. It is suitable for variable size of variable.
4. For creating such memory allocation we must use dynamic memory allocation functions such as malloc, calloc, realloc which are available in #include<stdlib.h> or #include<alloc.h>
5. In TurboC or TurboC++ compiler we must be declaration or initialization of variable must be first statement in the body of main.

**Note:-**For taking experience of this point please save C program file as .c

extension in TurboC++ compiler

1. We can use $ special character to start variable name in Java and PHP.
2. We must declared or initialize variable before access it in C language. See example to understand it.

Example:-1

int a=10;

printf(“a=%d”,a);

Above example is valid in C language.

Example:-2

printf(“a=%d”,a);

int a=10;

Above example is invalid in C language.

1. We can initialize variable without specifying data type in C++ by doing following

auto a=10;// Valid in C++ of C++11 standard.

since C++11 standard of C++. But this not allow in C.

1. We cannot define normal variable of void data type we must have pointer of void data type.
2. Memory for dynamic allocation is created in heap memory.
3. Constant variable value never changed. And constant variable must initialize.
4. Remember that variable is temporary memory. So we want to store value of variable then we must use file handling for doing this in C language and file handling as backend and file handling and database for doing this in C++ , Java and PHP.
5. Memory size of variable is depending on bit of compiler. This is because C language is not **Architecture Neural**.
6. Memory requires for integer, float and character variable is given below.

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr.no** | **Data type** | **Size in Bits** | **Size in Bytes** |
|  | Integer | 16 | 2 |
|  | Float | 32 | 4 |
|  | Character | 8 | 1 |
|  | Void | 8 | 1 |

**Table for 16 bits compiler**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr.no** | **Data type** | **Size in Bits** | **Size in Bytes** |
|  | Integer | 32 | 4 |
|  | Float | 32 | 4 |
|  | Character | 8 | 1 |
|  | Void | 8 | 1 |

**Table for 32 bits compiler**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr.no** | **Data type** | **Size in Bits** | **Size in Bytes** |
|  | Integer | 32 | 4 |
|  | Float | 32 | 4 |
|  | Character | 8 | 1 |

**Table for 64 bits compiler**

**Note:-**

1. The size of data type depends on how many bits compiler you have.
2. This is because C language is not **Architecture Neutral**.
3. For knowing size of variable we must use sizeof special operator.
4. We cannot finding size of void data type on TurboC and TurboC++ compiler.
5. For more understanding please run point no 14 program on different compiler.
6. We can calculate range of data type or find from which range variable store value by using this formula.

-2Number of bits for data type -1  to 2Number of bits -1  -1 For signed variable.

2Number of bits for data type  For unsigned variable.

Example:-1

Integer data type is of 2 bytes.

2 bytes=16 bit

Put 16 bit in above formula let us take view

-216-1 to 216-1-1

-32768 to 32767

So that the range of integer data type for 16 bits compiler is from

-32768 to 32767 for signed int.

Example:-2

Integer data type is of 2 bytes.

2 bytes=16 bit

Put 16 bit in above formula let us take view

216-1

65535

So that the range of integer data type for 16 bits compiler is from

65535 for unsigned int.

1. Understand and run this program on different compiler.

#include<stdio.h>

#include<conio.h>

main()

{

printf("Integer datatype is=%d",sizeof(int));

printf("\nFloat datatype is=%d",sizeof(float));

printf("\nChar datatype is=%d",sizeof(char));

printf("\nVoid datatype is=%d",sizeof(void));

printf("\nBool datatype is=%d",sizeof(\_Bool);

getch();

}

**Note**:- This program does not run on TurboC and TurboC++ compiler.

Originally created on 15/3/2020