Structure Padding:

Extra bytes are inserted between memory address which are allocated for other structure members while memory allocation.

EX:

struct Node

{

char c;

char ca;

short int dsa;

short int hk;

int a;

float b;

}st;

Sizeof(st) is 16

|  |  |  |
| --- | --- | --- |
| **Datatype** | **size** | **address** |
| Char | 1 | 4454844 |
| Char | 1 | 4454855 |
| Short int | 2 | 4454856 |
| Short int | 2 | 4454858 |
| Int | 4 | 4454862 |
| Float | 4 | 4454866 |

Memory address 4454860, 4454861 are left empty.

#pragma pack(1) is used for avoid the structure padding.

# Memory Layout:

## Text segment:

This field contains the executable program.

## Data segment:

2 types-Uninitialized data, Initialized data

### Uninitialized data:

It contain uninitialized global, static data that are initialized with zero

### Initialized data:

It contain initialized global, static and constant data that are initialized with non-zero

## Stack:

* + It is store the local variable of the function.
  + Instruction are performed in LIFO order
  + Execution is completed ,It is destroyed the stack memory

## Heap:

* + Dynamically allocated memory store in heap.
  + Allocate and free memory at any time

# Data types:

|  |  |  |
| --- | --- | --- |
| Data type | Bytes | Range |
| Char | 1 | -128 to 127 |
| Int | 2 | 215  to 215-1 |
| Float | 4 | 231 to 231-1 |
| Double | 8 | 263  to 263-1 |

## Modifier:

Change the range of the data type.

signed -values negative,0,positive number

unsigned – 0,positive value

long –increment the size of data type(int -2 bytes ,long int -4 bytes)

short – reduce the data type size (int -2 bytes, short int -1 byte)

## Qualifier:

const – can’t change the value once it define.

Volatile – the value is explicitly change by other program

# Pointer:

It is store the address of another variable.

int a=10;

int \*b=&a;

b is store the address of a.

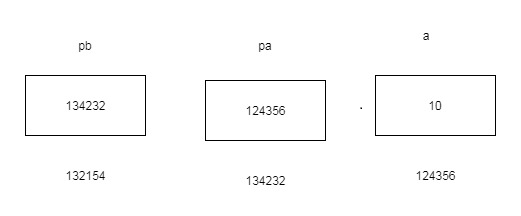


## Pointer to Pointer:

int a=10,\*pa,\*\*pb;

pa=&a;

pb=&pa;



## Pointer to function:

void sum(int a, int b)

{

printf("%d", a + b);

}

int main()

{

void(\*fun)(int a, int b);

fun = &sum;

fun(5, 9);

getchar();

return 0;

}

# Dynamic memory allocation:

malloc, calloc are used for allocate the dynamic memory.

## malloc:

It is return the pointer .It is not initialize the value.

char \*Ptr=(char \*) malloc(sizeof(char));

Randomly allocate the memory.

## calloc:

Ptr=(char \*) calloc (25,sizeof(char));

It allocates the block of memory and initializes the value as 0.

## realloc:

realloc is used for reallocate the memory.

Ptr=(char \*) realloc (Ptr,newsize);

free() is used for deallocate the memory.

# Storage classes:

Auto - scope within the block

Static - it don’t redeclare .scope entire program. It is not initializing then automatically initialize at 0.

Global – scope entire program ,it is not initializing then automatically initialize at 0.

Register – value store in the register memory. fast compilation

# Preprocessor:

## Macro:

#define –define the macro

Ex:

#define MAX 10

Int main()

{

Printf(“%d”,MAX);

}

In compilation the MAX is change to 10

i.e printf(“%d”,10);

#ifdef- if define do the process

Ex

#define MAX 10

Int main()

{

#ifdef MAX

Printf(“%d”,MAX);

#else

Printf(“%d”,MAX+1);

}

#ifndef - if macro not defines then do the process.

#undef –undefined the macro.