Calloc (Allocate and zero-initialize array) adalah salah satu fungsi untuk alokasi memori dengan elemen, yang tersedia pada bahasa pemrograman C dan juga pada bahasa pemrograman C++, Calloc tersedia pada header <stdlib.h> / <cstdlib>.

Calloc berfungsi untuk mengalokasi atau memesan memori bersama dengan elemen untuk sebuah array. Mengalokasi secara dinamis pada heap, dan pada masing-masing elemen mempunyai besar memori sebanyak byte yang disebutkan. Untuk bisa mengakses memori yang sudah di alokasikan kita bisa menggunakan operasi pointer. jika berhasil melakukan alokasi memori maka akan memberikan nilai berbentuk alamat kepada variabel pointer bersangkutan, Jika tidak berhasil melakukan alokasi maka akan member nilai “NULL” pada pointer. berbeda seperti malloc, pada saat alokasi atau pemesanan memori berhasil, masing-masing isi memori pada setiap elemen dilakukan inisialisasi yang berisi dengan NULL.

Array atau Larik adalah rangkaian elemen dari deretan memori yang dialokasikan saling berdekatan dengan elemen lain dan masing-masing memiliki jenis yang sama. dan Dinamis memori adalah suatu teknik alokasi memori yang memungkinkan memesan atau alokasi memori di saat program berjalan atau eksekusi program, bukan saat pembuatan atau penulisan program. Contoh kasus adalah, ketika pengguna program membutuhkan memori lebih untuk menyimpan data bukan dengan hanya menggunakan memori statis yang telah kita siapkan sebelumnya pada pembuatan program, maka teknik dinamis memori bisa kita manfaatkan pada kasus tersebut.

Bentuk umum penulisan / Syntax :

void\* calloc (size\_t num, size\_t size) ;

Parameter :

Num = jumlah block atau elemen yang akan dipesan.

Size = besar memori tipe data.

Contoh Penulisan :

varFoo = (int\*) calloc (5, sizeof(int)) ;

Return Value :

Jika fungsi calloc berhasil dalamalokasi maka calloc akan memberikan nilai berbentuk alamat kepada pointer dan jika gagal maka akan memberikan nilai NULL pada pointer.

Cara Pengaksesan :

Sama seperti pointer menunjuk ke variabel array, pointer akan menunjuk alamat pertama dari elemen array. begitu pula dengan cara pengaksesan pointer dengan variabel array dan pointer dengan array dinamis dengan calloc adalah sama. Untuk macam-macam cara megakses anda bisa mengunjungi artikel penjelasan “Pointer pada Array”

Perbedaan antara variabel array dengan array dinamis adalah jumlah banyaknya elemen akan ditentukan pada saat pembuatan atau penulisan program berbeda dengan array dinamis, array dinamis kita atau pengguna program dimungkinkan untuk menentukan jumlah elemen yang dibutuhkan saat program berjalan.

Contoh Program :

#include <stdio.h>

#include <stdlib.h>

#define MAX\_ARR 5 //Konstanta

using namespace std;

int main (){

int \*pointer = nullptr ; //Null Pointer

printf("pointer = %s \n\n", pointer); //Null Pointer

pointer = (int \*) calloc(MAX\_ARR, sizeof(int)); //Fungsi Calloc

for(int i=0;i<MAX\_ARR;i++){ //Loop for

printf("Masukan Angka pada Elemen ke- %d = ", i);scanf("%d", &pointer[i]);

}

printf("\n\nAngka Yang anda Masukan = ");

for(int i=0;i<MAX\_ARR;i++){ //Loop For

printf("\nElemen ke- %d = %d", i, pointer[i]);

}

free (pointer); //Melepas Memori

return 0;

}

Di atas merupakan contoh program yang menggunakan fungsi calloc. Pada contoh diatas cukup jelas bagaimana fungsi calloc digunakan.

Pada baris pertama, tepatnya pada “ int \*pointer = nullptr ; ” kita dirikan variabel pointer yang di inisialisasi dengan NULL, dapat kita butikan pada baris berikutnya “ printf("pointer = %s \n\n", pointer); ” yang menampilkan isi dari variabel pointer. pada pernyataan baris ketiga “ pointer = (int \*) calloc(MAX\_ARR, sizeof(int)); ” dicontohkan bagaimana penggunaan fungsi calloc. Di situ kita memesan atau mengalokasi memori dengan sejumlah elemen menjadikanya sebuah array, yaitu 5 elemen dengan menggunakan tipe data int.

Pada pernyataan-pernyataan berikutnya seperti biasanya apa yang kita lakukan pada penggunan Array, kita menggunakan pengulangan sebayak elemen yang telah dipesan dan memasukan nilai pada setiap elemen di dalamnya dengan menggunakan subscript “pointer[i]”. Begitu pula pada bagian keluaran.