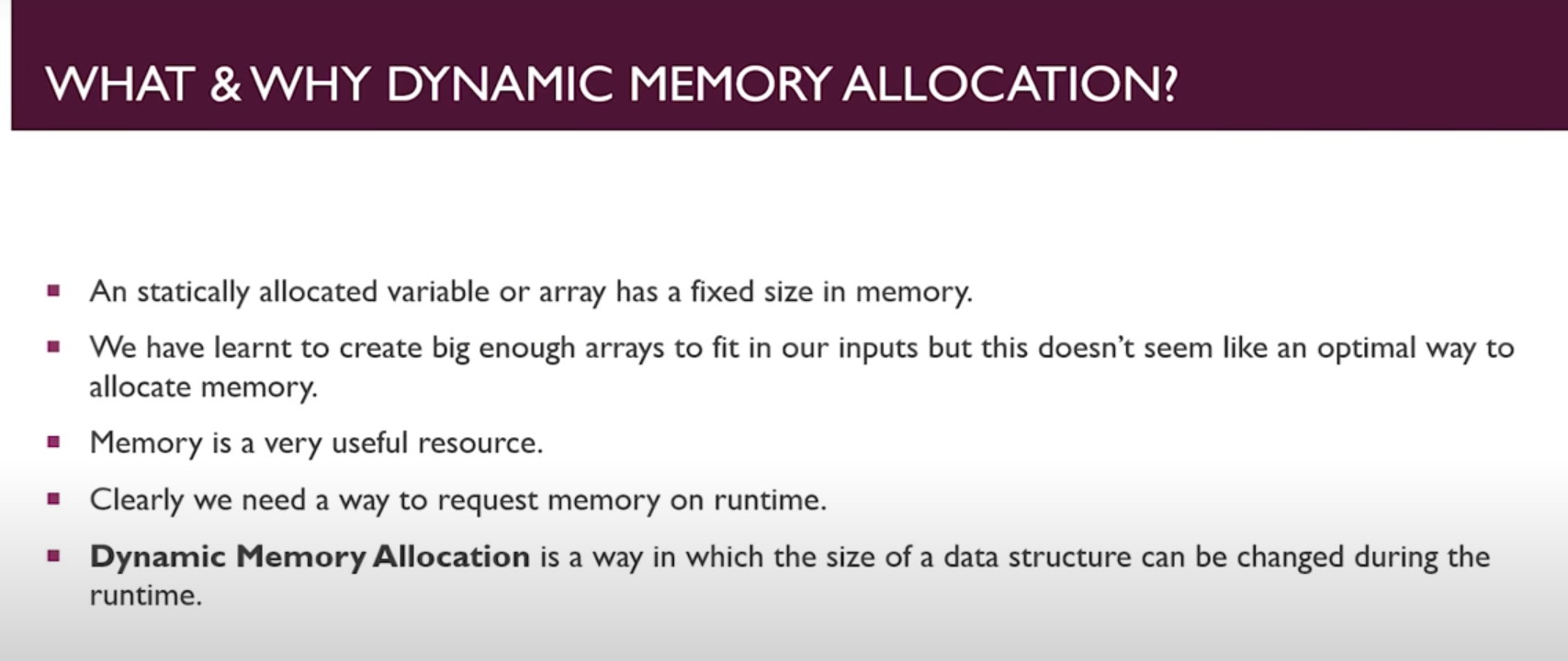
Content 33

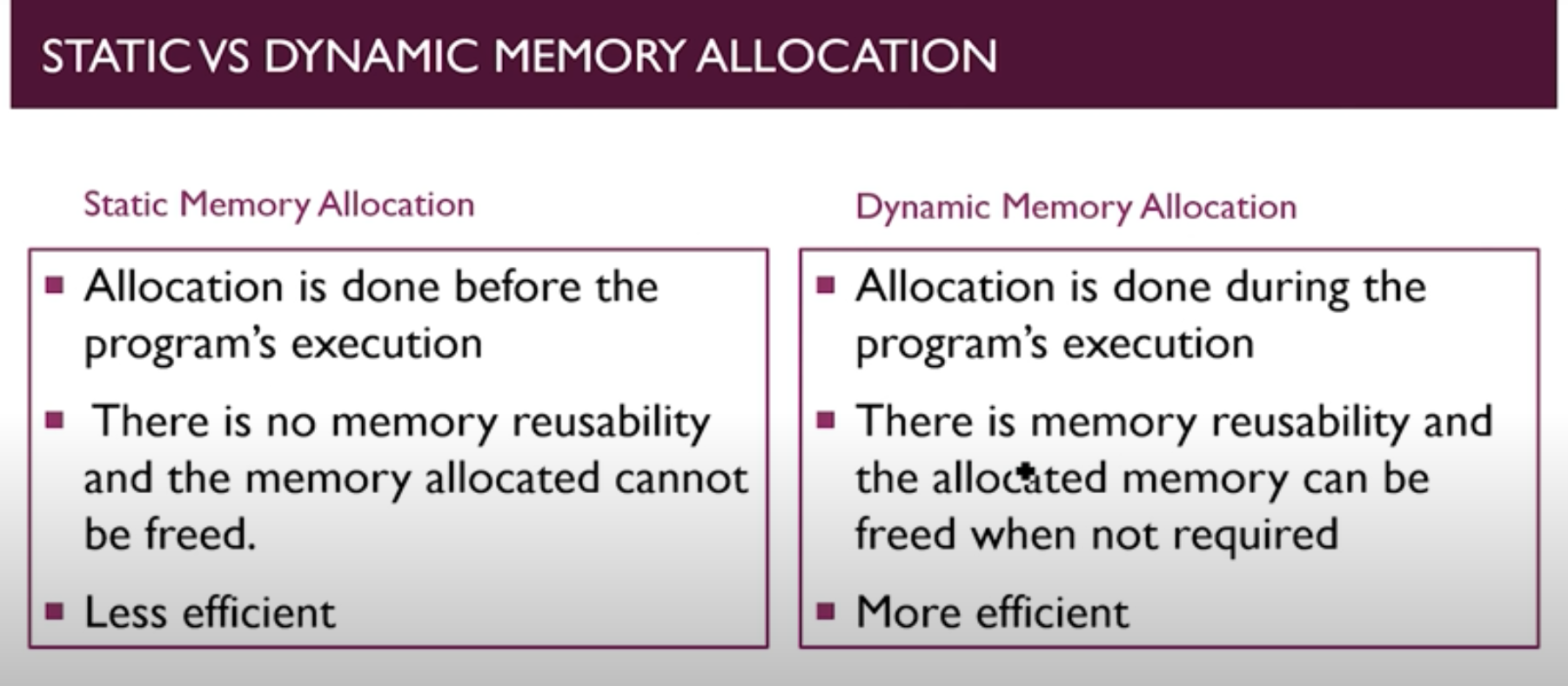
Memory Layout of C Programs

#### “Today we are going to learn about ****dynamic memory allocation****. This is one of the concepts that differentiate between a basic and a pro-level programmer”.

#### What is Dynamic memory?

Dynamic memory allocation is the process of allocation of memory space at the run time. We use the concept of dynamic memory allocation to reduce the wastage of memory, and it is the optimal way of memory allocation. To grasp the concept completely, we will see the memory layout of C programming.





**Memory Allocation In C Programs:**

1. Code (also called text segment). Data Segment. (Stores initialized data)

2.Static or Global Variable. BSS (Block Started by Symbol). (Stores initialized data)

3. Stack (which follows LIFO sys.)

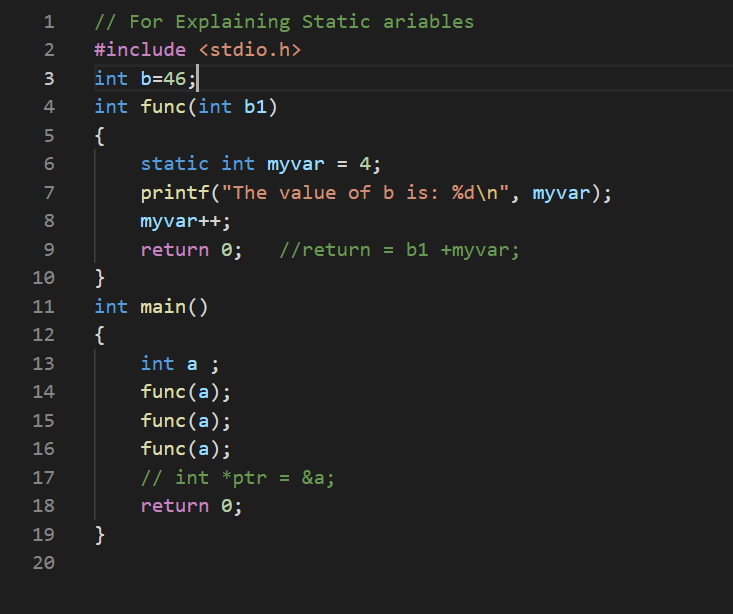
4. Heap. (Dynamic memory)

*“Using Heap is called Dynamic Memory Allocation”.*

**C Program: Memory Layout**

Showing a Program for giving more better Explanation of above Memory Allocation points.





Explanation of, How the program settle’s itself into Memory Layout?

1. The Whole code from line no. 1 to 20 will come in this **code segment or Text segment**
2. Global Variable of line 3 will come in **data Segment** of our **Global or Static Variable Segment** Because it is Initialized variable. ( but if it is uninitialized then it will come in **BSS** **segment** of our **Global or Static Variable segment.**)
3. The execution of program takes place from main body of program (from line no11) but as the program is passed to **stack**, the all variables( a to z variables) of main body will come in **stack** **frame** (list of storage which stores all the variables of that function) first and creates a **stack** **frame** of main then as main reaches to the func of line 14 **stack** **frame** stop itself and calls func and pushes func in itself(**stack** **frame**) and stores all Local variable in itself, and execution starts and while execution of func is going on main will wait until func returns (because stack follows LIFO system). Here in func Funtion I had also used an inbuilt printf function of stdio.h library of C then there will create an **stack** **frame** for printf fuction above func and first execute it while other (below ones which are func and main both are paused) then the below ones.

I am placing a figure 0.1 that how stack works? and also for better understanding of above content ;

**3.**

**2.**

**1.**

FIG 0.1

**STACK**

How the Stack Works?

**MAIN**

FUNC

PRINF

Stack Works on Last In First Out System.

Which means, here first entry taken by

Main then Func and then printf (at last).

So according to stack principle, last function

(printf) Will execute first while others will

Wait Until the execution of last get

completed.

So overall execution will take place as

1. Printf
2. Func
3. main

“finally stack over’s where first one gets returns, which in our case is main”.

**This is how our stack works.**

1. Heap:- It is Dynamic memory and using of Heap is called Dynamic memory allocation.

STACK

HEAP

“Generally Stack and Heap Grows in opposite directionin fig 0.2”.

If Stack and Heap meets then you had used too much memory

And your memory gets exhausted. Here your stack gets overflow.

**Stack Overflow:** Compiler allocates some space for the stack

part of memory which depends completely on Architecture

that how much it would reserved as Stack. Here stack is not a

small is good enough memory to manage program and it is completly in our hand which stores and relise function one by one.

But in any how you Exhuast it. But when can you do this? When write any recursive program in which function, as func1 in which you are calling that func1. This is **Stack Overflow**

**But when it is used by a program?**

We use Heap because;

1. There are Several limitation in stack (which is Static Memory allocation) like we can’t modify our storage after ones used or assigned, we can’t also free our storage ones our work is done.
2. Advantage of Heap is, it can be used be used fexibly by a programmer as per his needs.
3. Disadvantage of Heap is, if you make an mistake in program then it would increase the size of heap and it would not stop and your program will consume lot of memory

until you correct your mistake.

So it is like a power “Great power comes with great responsibility”.

**Code For deriving the above Memory allocation things:**

#include<stdio.h>

int main(int argc, char const \*argv[])

{

    printf("Hello");

    return 0;

}

This is simple code but I am want to show you memory allocation.

Go to Terminal do this;

**Output:**

PS C:\Users\Aamaan Satvilkar\Desktop\C cource> gcc main.c

PS C:\Users\Aamaan Satvilkar\Desktop\C cource> size .\a.exe

text data bss dec hex filename

8812 1556 1004 11372 2c6c .\a.exe

We have to type size before exe file and Simply we get this which shows Space occupation in bytes.