DATA STRUCTURES AND ITS APPLICATIONS

Vandana M L

Department of Computer Science and Engineering

DATA STRUCTURES AND ITS APPLICATIONS

Memory Allocation

Vandana M L

Department of Computer Science and Engineering

➢ Static Memory Allocation

➢ Dynamic Memory Allocation

- ➢ allocated by the compiler.
- ➢ Exact size and type of memory must be known at compile time
- ➢ Memory is allocated in stack area

int b;

int c[10] ;

➢ Memory allocated can not be altered during run time as it is allocated during compile time

➢ This may lead to under utilization or over utilization of memory

➢ Memory can not be deleted explicitly only contents can be overwritten

➢ Useful only when data size is fixed and known before processing

➢ Dynamic memory allocation is used to obtain and
   release memory during program execution.
➢ It operates at a low-level
➢ Memory Management functions are used for allocating
   and deallocating memory during execution of program
➢ These functions are defined in "stdlib.h"

Dynamic Memory Allocation Functions:
  –   Allocate memory - malloc(), calloc(), and realloc()
  –   Free memory - free()

Dynamic Memory Allocation Functions: malloc()

To allocate memory use

void *malloc(size_t size);

- Takes number of bytes to allocate as argument.

- Use sizeof to determine the size of a type.

- Returns pointer of type void *. A void pointer may be assigned to any pointer.

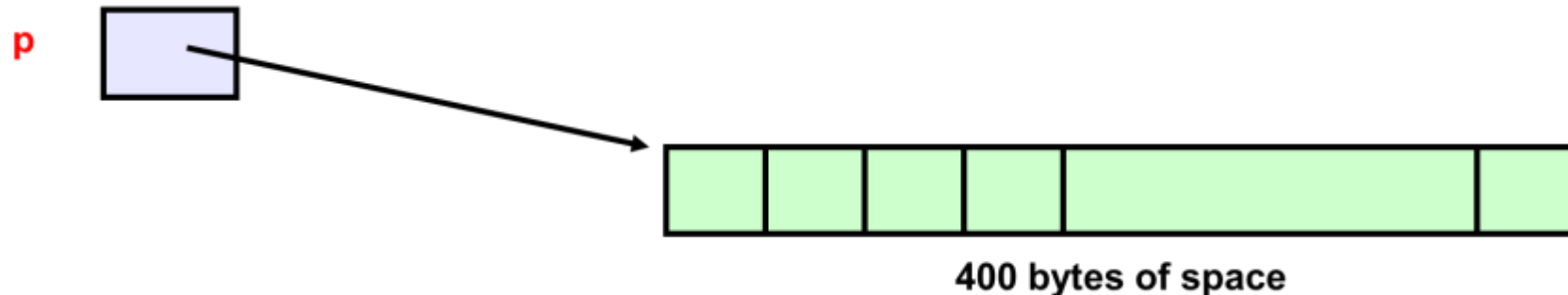- If no memory available, returns NULL.

To allocate space for 100 integers:

int *p;

if ((p = (int*)malloc(100 * sizeof(int))) == NULL){

    printf("out of memory\n");

    exit();

    }

- Note we cast the return value to int*.
- Note we also check if the function returns NULL.

p

400 bytes of space

- cptr = (char *) malloc (20);

    Allocates 20 bytes of space for the pointer cptr of type char

- sptr = (struct stud *) malloc(10*sizeof(struct stud));

    Allocates space for a structure array of 10 elements. sptr points
    to a structure element of type struct stud

**Always use sizeof operator to find number of bytes for a data type,
    as it can vary from machine to machine**

- malloc always allocates a block of contiguous bytes
  - The allocation can fail if sufficient contiguous memory space is not available
  - If it fails, malloc returns NULL

```
if  ((p = (int *) malloc(100 * sizeof(int))) == NULL)
  {
        printf ("\n Memory cannot be allocated");
        exit();
  }
```

The n integers allocated can be accessed as *p, *(p+1), *(p+2),…, *(p+n-1) or just as p[0], p[1], p[2], …,p[n-1]

To release allocated memory use

free(ptrvariable)

- Deallocates memory allocated by malloc().

- Takes a pointer as an argument.

e.g.

free(newPtr);

Dangling pointer

Memory Leak

Similar to malloc(),

But allocated memory space are zero by default..

calloc() requires two arguments –

void *calloc(size_t nitem, size_t size);

Example

int *p;

p=(int*)calloc(100,sizeof(int));

returns a void pointer if the memory allocation is successful, else it'll return a NULL pointer.

Reallocate a block

Two arguments

- Pointer to the already allocated block

- Size of new block

int *ip;

ip = (int*)malloc(100 * sizeof(int));

. . .

/* need twice as much space */

ip = (int*)realloc(ip, 200 * sizeof(int));

Memory Allocation

➢ Static Memory allocation
➢ Dynamic memory allocation

Apply the concepts to implement C program for the following problem statement

➢ Multiply two matrices . Allocate the memory for the matrices dynamically

THANK YOU

Vandana M L

Department of Computer Science & Engineering

vandanamd@pes.edu

+91 7411716615