

DigiPen

Dynamic Memory

Practice

© 2020, DigiPen Institute of Technology. All Rights Reserved

Static Memory

- Most memory that we use is static (AKA automatic).
 - The programmer didn't have to worry about finding available memory; the compiler did it for you.
 - You also didn't have to worry about releasing the memory when you were finished with it; it happened automatically.
 - Static memory allocation is easy and effortless, but it has limitations.

```
void foo() {  
    int n = 0; /* static int */  
    printf("%d", n);  
}
```

Dynamic Memory

- Dynamic memory allocation is under complete control of the programmer.
 - This means that you will be responsible for allocating and de-allocating memory.
 - Failing to understand how to manage the memory yourself will lead to programs that behave badly (crash).

C

- In C, the two most used functions for dynamic memory management are malloc and free.

```
void* malloc( size_t size );  
/* Allocate a block of memory */  
  
void free( void* pointer );  
/* Deallocate a block of memory */
```

- To use malloc and free, you must include:

```
#include <stdlib.h>
```

C Example

Run

```
#include <stdio.h>  
#include <stdlib.h>  
#define SIZE 10  
int main() {  
    /* allocate */  
    int i;  
    int *pi = NULL;  
    pi = (int*)malloc(SIZE*sizeof(int));  
    /* check for valid pointer */  
    if (!pi) {  
        printf("Failed.\n");  
        return -1;  
    }  
    /* set */  
    for (i = 0; i < SIZE; ++i)  
        pi[i] = i;  
    /* output */  
    for (i = 0; i < SIZE; ++i)  
        printf("%i", pi[i]);  
    /* deallocate */  
    free(pi);  
    return 0;  
}
```

0123456789

★ C++'s new/new[]

- C++ supports malloc/free functions and also has two operators new and delete that perform the task of allocating and freeing the memory in a better and easier way.
- The new operator denotes a request for memory allocation on the heap.
- If sufficient memory is available, new operator initializes the memory and returns the address of the newly allocated and initialized memory to the pointer variable.

```
int *p = new int;
```

- We can also initialize the memory using new operator:

```
float *q = new float(75.25);
```

- new operator is also used to allocate a block (an array) of memory:

```
int *pi = new int[10];
```

★ C++'s delete/delete[]

- For dynamically allocated memory it is programmers responsibility to deallocate memory when no longer needed.
 - If programmer doesn't deallocate memory, it causes memory leak (memory is not deallocated until program terminates).
- To deallocate dynamically allocated memory, programmers are provided delete operator by C++ language.

```
delete p; // for a variable or  
          // a structure pointed by p  
delete[] a; // for an array a
```

★ C++ Example

```
Run
#include <iostream>
#include <new>
int main() {
    const int SIZE = 10;
    /* allocate */
    int *pi = new int[SIZE];
    /* set */
    for (int i = 0; i < SIZE; ++i)
        pi[i] = i;
    /* output */
    for (int i = 0; i < SIZE; ++i)
        std::cout << pi[i];
    /* deallocate */
    delete[] pi;
    return 0;
}
```

```
0123456789
```

By signing this document you fully agree that all information provided therein is complete and true in all respects.

Responder sign: