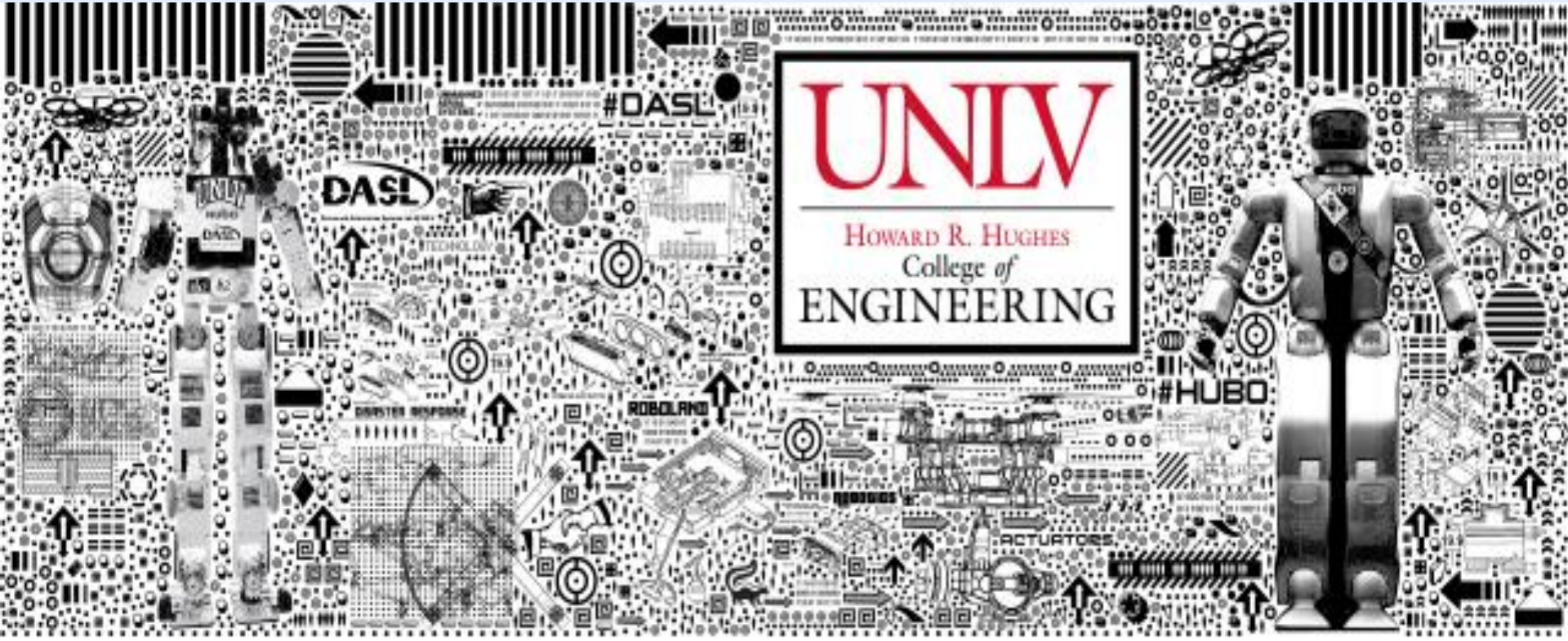
UNIVERSITY OF NEVADA LAS VEGAS
DRONES AND AUTONOMOUS SYSTEMS LAB (DASL@UNLV)



C PROGRAMING / LINUX [DASL-100]

WEEK 4 [Section 7]

INSTRUCTOR: JEAN CHAGAS VAZ

Wednesday, June 14, 2017, 10:37

➢ C Program to Access Elements of an Array Using Pointer

➢This program declares the array of five element and the elements of that array are accessed using pointer.

```c
#include <stdio.h>

int main()
{
    int data[5], i;
    printf("Enter elements: ");

    for(i = 0; i < 5; ++i)
       scanf("%d", data + i);

    printf("You entered: \n");
    for(i = 0; i < 5; ++i)
        printf("%d\n", *(data + i));

    return 0;
}
```

```
Enter elements: 1
2
3
5
4
You entered:
1
2
3
5
4
```

Source: programiz.com

Wednesday, June 14, 2017, 10:42

➢ C Call by Reference: Using pointers

➢When a pointer is passed as an argument to a function, address of the memory location is passed instead of the value.

➢This is because, pointer stores the location of the memory, and not the value.

```
Number1 = 10
Number2 = 5
```

```c
/* C Program to swap two numbers using pointers and function. */
#include <stdio.h>
void swap(int *n1, int *n2);

int main()
{
    int num1 = 5, num2 = 10;

    // address of num1 and num2 is passed to the swap function
    swap( &num1, &num2);
    printf("Number1 = %d\n", num1);
    printf("Number2 = %d", num2);
    return 0;
}

void swap(int * n1, int * n2)
{
    // pointer n1 and n2 points to the address of num1 and num2 respectively
    int temp;
    temp = *n1;
    *n1 = *n2;
    *n2 = temp;
}
```

Source: programiz.com

➢ The address of memory location num1 and num2 are passed to the function swap and the pointers *n1 and *n2 accept those values.

➢So, now the pointer n1 and n2 points to the address of num1 and num2 respectively.

➢When, the value of pointers are changed, the value in the pointed memory location also changes correspondingly.

➢Hence, changes made to *n1 and *n2 are reflected in num1 and num2 in the main function.

➢This technique is known as Call$_3$ by Reference in C programming.

➢ C Program Swap Numbers in Cyclic Order Using Call by Reference

➢This program takes three integers from the user and swaps them in cyclic order using pointers.

```c
#include<stdio.h>
void cyclicSwap(int *a,int *b,int *c);

int main()
{
    int a, b, c;

    printf("Enter a, b and c respectively: ");
    scanf("%d %d %d",&a,&b,&c);

    printf("Value before swapping:\n");
    printf("a = %d \nb = %d \nc = %d\n",a,b,c);

    cyclicSwap(&a, &b, &c);

    printf("Value after swapping:\n");
    printf("a = %d \nb = %d \nc = %d",a, b, c);

    return 0;
}
void cyclicSwap(int *a,int *b,int *c)
{

    int temp;

    // swapping in cyclic order
    temp = *b;
    *b = *a;
    *a = *c;
    *c = temp;
}
```

```
Enter a, b and c respectively: 1
2
3
Value before swapping:
a = 1
b = 2
c = 3
Value after swapping:
a = 3
b = 1
c = 2
```

Source: programiz.com

➢ C Dynamic Memory Allocation

➢In C, the exact size of array is unknown until compile time, i.e., the time when a compiler compiles your code into a computer understandable language. So, sometimes the size of the array can be insufficient or more than required.

➢Dynamic memory allocation allows your program to obtain more memory space while running, or to release it if it's not required.

➢In simple terms, Dynamic memory allocation allows you to manually handle memory space for your program.

➢Although, C language inherently does not have any technique to allocate memory dynamically, there are 4 library functions under "stdlib.h" for dynamic memory allocation.

Function	Use of Function
malloc()	Allocates requested size of bytes and returns a pointer first byte of allocated space
calloc()	Allocates space for an array elements, initializes to zero and then returns a pointer to memory
free()	deallocate the previously allocated space
realloc()	Change the size of previously allocated space

Source: programiz.com

Wednesday, June 14, 2017, 10:47

➢ C Program to Find Largest Number Using Dynamic Memory Allocation

➢In this program, you'll learn to use calloc() function to allocate the memory dynamically to find the largest element.

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

int main()
{
    int i, num;
    float *data;

    printf("Enter total number of elements(1 to 100): ");
    scanf("%d", &num);

    // Allocates the memory for 'num' elements.
    data = (float*) calloc(num, sizeof(float));

    if(data == NULL)
    {
        printf("Error!!! memory not allocated.");
        exit(0);
    }

    printf("\n");

    // Stores the number entered by the user.
    for(i = 0; i < num; ++i)
    {
        printf("Enter Number %d: ", i + 1);
        scanf("%f", data + i);
    }

    // Loop to store largest number at address data
    for(i = 1; i < num; ++i)
    {
        // Change < to > if you want to find the smallest number
        if(*data < *(data + i))
            *data = *(data + i);
    }

    printf("Largest element = %.2f", *data);

    return 0;
}
```

```
Enter total number of elements(1 to 100): 10

Enter Number 1: 2.34
Enter Number 2: 3.43
Enter Number 3: 6.78
Enter Number 4: 2.45
Enter Number 5: 7.64
Enter Number 6: 9.05
Enter Number 7: -3.45
Enter Number 8: -9.99
Enter Number 9: 5.67
Enter Number 10: 34.95
Largest element: 34.95
```

Source: programiz.com

Wednesday, June 14, 2017, 10:49

- **To do List**
  - What is the difference between (*i and i*) based on the pointers perspective
  - Create a program application that uses a third degree pointer reference (e.g ***i)(DUE NEXT SECTION)
  - Research and write a brief summary on the types of "Dynamic Memory Allocation" applied to C and C++ Language [Make sure to give two examples] (DUE NEXT SECTION)

Source: programiz.com