**Programming Fundamentals Lecture 11  
Output Finding Questions**

**Question 1:**

The precedence of prefix ++ or -- is same as of dereference operator '\*' and associate right to left

postfix ++ or -- has a priority higher than both prefix ++ and dereference operator '\*'.

Now compare the followings:

**1.** \*ptr++ // same as \*(ptr++) // means "increment ptr to go to the element after the one it points at, then dereference its old value"

int main() {

char arr[] = "Hello World";

char \*ptr = arr;

\*ptr++;

cout << \*ptr;

return 0;

}

**Output: e**

**2.**++\*ptr //same as ++(\*ptr) // means "increment the value pointed at by ptr,"

int main() {

char arr[] = "hello World";

char \*ptr = arr;

++\*ptr;

cout << \*ptr;

return 0;

}

**Output: i**

**3.**\*++ptr // same as \*(++ptr) // increment the ptr to go to the element after the one it points at, and dereference the old value of it

int main() {

char arr[] = "Hello World";

char \*ptr = arr;

\*++ptr;

cout << \*ptr;

return 0;

}

**Output: e**

**Question 2:**

int main()

{

char \* a1 = "0123456789";

int a2[10] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};

char \* ptr1 = &a1[3];

int \* ptr2 = &a2[3];

cout << "ptr1[0] = " << ptr1[0] << ", ptr1[6] = " <<

ptr1[6] << endl;

cout << "ptr2[0] = " << ptr2[0] << ", ptr2[6] = " <<

ptr2[6] << endl;

return 0;

}

**Output result:**

ptr1[0] = 3,

ptr1[6] = 9

ptr2[0] = 3,

ptr2[6] = 9

**Question 3:**

int main()

{

char \* temp1 = "abcdefg";

char temp2[20] = "abcdefg";

char temp3[30] = "abcdefg";

cout << "Size of a char \* is " << sizeof temp1

<< ", size of char[20] is " << sizeof temp2

<< ", size of char[30] is " << sizeof temp3 << endl;

return 0;

}

**Output:**

Size of a char \* is 4, size of char[20] is 20, size of char[30] is 30

**Question 3:** Is the NULL pointer same as uninitialized pointer?

**Question 4:**  Draw the memory diagram of the following code:

int main()

{

int \*p1,\*p2,i=25;

p1=&i;

p2=&i;

p1=p2;

p2=p1;

return 0;

}

**Question 6:**

void print (int \*p1,int \*p2)

{

while(p1<=p2)

{

cout<<\*p1;

p1++;

}

}

int main()

{

int a[10]={2,3,6,8,10,16,17};

print( &a[1],&a[5]);

return 0;

}

**PROGRAMMING FUNDAMENTALS  
LECTURE 11**

**Question 7:** Write a function to copy an array of size size1 named arr1 into array of size size2=size1-1 named arr2, by skipping the last element of arr1.

Use this function prototype:

void copy(int \*arr1 ,int size1,int\*arr2,int size2);

Use this function call:

If source is an integer pointer pointing an array of size1 and destination is an integer pointer pointing an array of size2 all initialized by user in main.   
copy (source , size1,destination,size2 );

**Question 8:** Write a function to copy an array of size size1 named arr1 into array of size size2=size1-1 named arr2, by skipping the last element. Write another function that will delete the arr1 once copied into arr2 and assign/update arr1 the same address pointed to by arr2. Also update the size1 to size2.  
***Note: This concept is known as shrinking/reducing size of dynamically allocated arrays***

**Use these function prototype:**

**void copy(int \*arr1 ,int size1,int\* arr2,int size2);  
void shrink(int \*&arr1,int &size1);  
Make call of copy function inside shrink instead of main.  
Create arr2 pointer inside shrink instead of main.**

**Use this function call:**

**If source is an integer pointer pointing an array of size1 all initialized by user in main.   
shrink (source, size1);**

**Question 9:** Write a function to copy an array of size size1 named arr1 into array of size size2=size1+1 named arr2, by leaving blank the last element of arr2. Write another function that will delete the arr1 once copied into arr2 and assign/update arr1 the same address pointed to by arr2. Also update the size1 to size2.  
***Note: This concept is known as re-growing/increasing size of dynamically allocated arrays***

**Use these function prototype:  
void copy(int \*arr1 ,int size1,int\* arr2,int size2);  
void regrow(int \*&arr1,int &size1);  
Make call of copy function inside regrow instead of main.  
Create arr2 pointer inside regrow instead of main.**

**Use this function call: regrow( source, size1);**