Q1:

## There need to be brackets around ptr+1. The loop condition needed to be reversed to make it so that arr[0] printed first.

**int** main() {

**int** arr[3] = { 5, 10, 15 };

**int**\* ptr = arr; //ptr = arr[0]

\*ptr = 30; // set arr[0] to 30

\*(ptr + 1) = 20; // set arr[1] to 20

ptr += 2; //ptr = arr[2]

ptr[0] = 10; // set arr[2] to 10

ptr = arr;

**while** (ptr <= arr + 2)

{

cout << \*ptr << endl; // print values

ptr++;

}

}

## The functions works perfectly but since pToMax is passed by value, its actual value doesnt change so the pointer never moves from position nums[0]. I just passd it by reference.

**void** findMax(**int** arr[], **int** n, **int**\*& pToMax)

{

**if** (n <= 0)

**return**; // no items, no maximum!

pToMax = arr;

**for** (**int** i = 1; i < n; i++)

{

**if** (arr[i] > \*pToMax)

pToMax = arr + i;

}

}

**int** main()

{

**int** nums[4] = { 5, 3, 15, 6 };

**int**\* ptr = &nums[0];

findMax(nums, 4, ptr);

cout << "The maximum is at address " << ptr << endl;

cout << "It's at position " << ptr - nums << endl;

cout << "Its value is " << \*ptr << endl;

}

## ptr is uninitialised when used here so I created a variable num and set ptr to point to it.

**void** computeCube(**int** n, **int**\* ncubed)

{

\*ncubed = n \* n \* n;

}

**int** main()

{

**int** num = 5;

**int**\* ptr = &num;

computeCube(num, ptr);

cout << "Five cubed is " << \*ptr << endl;

}

## We cant compare c strings using the ‘!=’ operator as in this case, the strings itself dont get compared but instead the pointers str1 and str2 get compared which will show not equal even when the chars of the strings are equal. We can fix this by comparing the where the pointer points to (the actual characters) instead.

**bool** strequal(**const** **char** str1[], **const** **char** str2[]) {

**while** (\*str1 != 0 && \*str2 != 0) // zero bytes at ends

{

**if** (\*str1 != \*str2) // compare corresponding characters

**return** **false**;

str1++; // advance to the next character

str2++;

}

**return** \*str1 == \*str2; // both ended at same time?

}

**int** main()

{

**char** a[15] = "Chang";

**char** b[15] = "Zhang";

**if** (strequal(a,b))

cout << "They're the same person!\n";

}

## The array ‘anArray[100]’ is declared inside a function so when other functions or the mainroutine try to access this array it causes errors and doesnt do as intended.

2.

1. double\* cat;
2. double mouse[5];
3. cat = &mouse[4];
4. \*cat = 25;
5. \*(mouse + 3) = 54;
6. cat -= 3;
7. cat[1] = 42;
8. cat[0] = 17;
9. bool d = (cat == mouse);
10. bool b = (\*cat == \*(cat + 1));

3.

**double** mean(**const** **double**\* scores, **int** numScores)

{

**double** tot = 0;

**for** (**int** k = 0; k < numScores; k++){

tot += \*(scores + k);

}

**const** **char**\* findTheChar(**const** **char** \*str, **char** chr)

{

**for** (**int** k = 0; \*(str + k) != 0; k++){

**if** (\*(str + k) == chr)

**return** (str + k);

}

**return** **nullptr**;

}

**const** **char**\* findTheChar(**const** **char** \*str, **char** chr)

{

**for** (; \*str != 0; str++){

**if** (\*str == chr)

**return** str;

}

**return** **nullptr**;

}

4. Output is:

3

4

79

-1

9

22

19

The pointer ptr starts at array[0] as function maxwell returned that. The next line of code then assigns -1 to position 0 in the array making it -1, 3, 4, 17, 22, 19. The pointer is then moved forward 2 positions to position 2 of the array. Then 9 is assigned to position 3 of the array making it -1, 3, 4, 9, 22, 19. 79 is then assigned to position 1 of the array making it -1, 79, 4, 9, 22, 19. The first output statement is the position of ptr in the array subtracted from the 5th position on the array which gives us 5 - 2, thus the first line outputted is 3. The first swap function doesnt actually swap the values inside the arrays but just the pointers. The second swap function swaps the values at pos 2 and 0 making the array 4, 79, -1, 9, 22, 19. This modified array is then just printed line by line using the for loop.

5.

**void** removeS(**char**\* str){

**for** (; \*str != 0; str++){

**if** (\*str == 's' || \*str == 'S'){

**while** (\*str == 's' || \*str == 'S')

**for** (**char**\* ptr = str; \*ptr != 0; ptr++){

\*ptr = \*(ptr+1);

}

}

}

}