Project 6

Brian Tagle

1a. The main problem with this program is that the increment for ptr in the loop comes before the cout line. Also, the values are printed in the wrong order, so I changed the order in which the values are assigned.

int main()

{

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

int\* ptr = arr;

\*ptr = 10; // set arr[0] to 10

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

ptr += 2;

ptr[0] = 30; // set arr[2] to 30

while (ptr >= arr)

{

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

ptr--;

}

}

1b. The return type is void and the pointer is only being changed in the function and changes to the pointer are not reflected in the pointer parameter after the function terminates. We must change the pointer parameter to be a 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;

}

}

1c. The pointer is not initialized to any value and cout cannot print a pointer that points to nothing. The function changes that value that the pointer is pointing too, but the pointer is not pointing to anything before the fix.

int main()

{

int n = 5;

int\* ptr = &n;

computeCube(n, ptr);

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

}

1d. The original function is comparing pointers to two different char arrays. The problem is they are comparing two pointers and not the objects they point to and they also are trying to increment the array pointers which you cannot do. I fixed this by making new pointers pointing at the char arrays so that I can easily compare and increment them up the array to check the chars in the array

bool strequal(char str1[], char str2[])

{

char \*ptr1 = &str1[0];

char \*ptr2 = &str2[0];

while (\*ptr1 != 0 && \*ptr2 != 0)

{

if (\*ptr1 != \*ptr2) // compare corresponding characters

return false;

ptr1++; // advance to the next character

ptr2++;

}

return \*ptr1 == \*ptr2 ; // both ended at same time?

}

1e. The array created in the function getPtrToArray successfully makes the pointer in the main function point to the location of the first element in the created array, however this location and all other locations in the array contain garbage values after the function getPtrToArray terminates because the array is a local variable in the function and the values in are thrown out after the function runs.

2.

//a

double\* cat = 0;

//b

double mouse[5];

//c

cat = &mouse[4];

//d

\*cat = 25;

//e

\*(mouse + 3) = 42;

//f

cat -= 3;

//g

cat[1] = 54;

//h

cat[0] = 27;

//i

bool b = (\*cat == \*(cat + 1));

//j

bool d = (cat == mouse);

3a.

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

{

double tot = 0;

for (int i = 0; i< numScores; i++)

{

tot += \*(scores+i);

}

return tot / numScores;

}

3b.

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;

}

3c.

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

{

for (str; \*str != 0; str = str + 1) {

if (\*str == chr)

return str;

}

return nullptr;

}

4. { 5, 3, 4, 17, 22, 19 }; is the starting array, the first function called creates a pointer pointing at the first value in the array and changes the first value to -1. The pointer is incremented up twice so it is pointing at 4. Then the pointer sets the value after 4 equal to 9. So now 17 is 9. Then the second value of the array is set to 79. The new array is

{-1, 79, 4, 9, 22, 19}.

The first line of output subtracts the address of the 6th element from the address of the 3rd element which are three spaces apart in computer memory so **3 is printed.**

Swap 1 does nothing except make pointer point to different places but then the pointers are thrown out anyway.

Swap 2 swaps the integers at position 0 and position 2 so now the array is

{-1, 79, 4, 9, 22, 19}.

Then the array is printed

**4**

**79**

**-1**

**9**

**22**

**19**

5.

void removeS(char\* str) {

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

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

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

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

}

ptr = str;

}

}

}