1.

a.

int main()

{

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

int\* ptr = arr;

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

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

ptr += 2;

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

ptr-=2;

while (ptr < &arr[3])

{

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

ptr++;

}

}

b.

Explain: In the original code, pToMax is not pass by reference, so ptr in the main function will not change since pToMax is a copy of ptr. findMax function only changes pToMax, a copy of ptr, but not ptr itself. Thus, we need to add & as int\*& pToMax to pass 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;

}

}

c.

explain: ptr has not been initialized, so we need create int x that ptr points to.

int main()

{

int x;

int\* ptr=&x;

computeCube(5, ptr);

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

}

d.

Explain: The problem is in the original code, the function compares the pointer. However, we are supposed to compare the char that pointers point to but not pointer, so we need to put \* in fount of str1 and str2.

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?

}

e.

The function tries to access local array initialized in the function getPtrToArray(), which is undefined behavior. Local variable can no longer be accessed outside of the function.

2.

int main(){

double\* cat;//a

double mouse[5];//b

cat = &mouse[4];//c

\*cat = 25;//d

\*(mouse+3)=54;//e

cat=cat-3;//f

(cat-1)[2]=17;//g

cat[0]=42;//h

bool d= (cat==mouse);//i

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

}

3.

a.

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

{

const double\* ptr = scores;

double tot = 0;

int x = 0;

while (x<numScores)

{

tot += \*(ptr+x);

x++;

}

return tot/numScores;

}

b.

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;

}

c.

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

while (\*str != chr){

str++;

if (\*str == '\0'){

return nullptr;

}

}

return str;

}

4.

int main()

{

int array[6] = { 5, 3, 4, 17, 22, 19 };//create an array with 6 elements

int\* ptr = maxwell(array, &array[2]);//initialize a pointer ptr to the return of function Maxwell, which is a &array[0], since array[0]>array[2]

\*ptr = -1;//assign array[0] to -1

ptr += 2;//assign ptr pointing to array[2]

ptr[1] = 9;//assign array[3] to be 9

\*(array+1) = 79;// assign array[1] to be 79

cout << &array[5] - ptr << endl;//print the difference between the pointer to array[5] and ptr, which is the pointer to array[2]. Thus, it will output 5-2=3

swap1(&array[0], &array[1]);//swap1 intend to exchange pointers &array[0] and &array[1], but it will not change the pointer in the main function, since it is not pass by reference.

swap2(array, &array[2]);// swap2 intend to exchange the value of array[0] and array[2], so array[0] will become4 and array[2] will become -1.

for (int i = 0; i < 6; i++)

cout << array[i] << endl;//print all elements in current array

}

Thus, the final output is

3

4

79

-1

9

22

19

5.

void removeS(char\* cString){

char\* newcString=cString;

while (\*newcString != '\0'){

newcString++;

}

while (newcString >= cString){

if (\*newcString != 's' && \*newcString != 'S'){

newcString--;

}

else

{

while (\*newcString != '\0'){

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

newcString++;

}

}

}

}