1. **a.**

int main()

{

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

int\* ptr = arr;

\*ptr = 10; // set arr[0] to 30

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

ptr[2] = 30; // set arr[2] to 10

ptr += 2;

while (ptr >= arr)

{

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

ptr--;

}

}

**b.**

/\*

This function does not work because the pointer pToMax is not passed into the function findMax by reference. Therefore, the findMax could not modify the actual pointer declared in the main. We can fix this by changing the parameter of the function so that the pToMax is passed by reference.

\*/

//corrected code:

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;

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;

}

**c.**

/\*

The main function does not work because the pointer ptr declared in the main function is not initialized as does not actually point to an integer value. Therefore, the function computeCube could not modify the value that the pointer is pointed.

\*/

//corrected code:

void computeCube(int n, int\* ncubed)

{

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

}

int main()

{

int x;

int\* ptr = &x;

computeCube(5, ptr);

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

}

**d.**

/\*

The function strequal did not work because it was trying to compare pointers to the characters of the C strings instead of the actual characters of the C strings.

\*/

//corrected code:

// return true if two C strings are equal

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

{

while (\*str1 != 0 && \*str2 != 0)

{

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] = "Zhao";

char b[15] = "Zhou";

if (strequal(a, b))

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

}

**e.**

Because the array anArray is only declared within the local scope of the function getPtrToArray, so as getPtrToArray returns the pointer to anArray, anArray itself no longer exists and cannot be accessed from the main function.

1. **a.** double\* cat;

**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] = 17;

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

**j.** bool d = (cat == mouse);

1. **a.**

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

{

const double\* ptr = scores;

double tot = 0;

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

{

tot += ptr[i];

}

return tot / numScores;

}

**b.**

// This function searches through str for the character chr.

// If the chr is found, it returns a pointer into str where

// the character was first found, otherwise nullptr (not found).

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)

{

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

if (\*i == chr)

return i;

return nullptr;

}

**int main()**

**{**

**int array[6] = { 5, 3, 4, 17, 22, 19 };**

**int\* ptr = maxwell(array, &array[2]);**

/\*

Pointers to array[0] and array [2] are passed into maxwell. The pointer that points to the larger value will be returned to become the value of ptr, which means ptr now points to array[0].

\*/

**\*ptr = -1;**

/\*

The value ptr points to is set to -1. This means array[0] now equals to -1.

Current value of array = {-1, 3, 4, 17, 22, 19}

\*/

**ptr += 2;**

/\*

ptr was moved forward by 2 indexes, it now points to array[2].

\*/

**ptr[1] = 9;**

/\*

The value one index forward of ptr is set to 9, which means array[3] now equals to 9.

Current value of array = {-1, 3, 4, 9, 22, 19}

\*/

**\*(array + 1) = 79;**

/\*

The value one index forward of array is set to 79, which means array[1] now equals to 79.

Current value of array = {-1, 79, 4, 9, 22, 19}

\*/

**cout << &array[5] - ptr << endl;**

/\*

Prints the result of the pointer to array[5] minus ptr.

&array[5] – ptr = &array[5] - &array[2] = 3

Print output:

3

\*/

**swap1(&array[0], &array[1]);**

/\*

This function attempts to modify the pointers passed in. However, the pointers are not passed in by reference, so they cannot by modified by this function. This line does nothing.

\*/

**swap2(array, &array[2]);**

/\*

This function swaps the values at the locations these two pointers point to. Values at array[0] and array[2] are swapped.

Current value of array = {4, 79, -1, 9, 22, 19} \*/

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

**cout << array[i] << endl;**

/\*

This for loops prints out every element of the array, one value in each line.

Print output:

4

79

-1

9

22

19

\*/

**}**

Complete print output：

3

4

79

-1

9

22

19

1. void removeS (char\* msg)

{

for(; \*msg != '\0'; msg++){

while(\*msg == 'S' || \*msg == 's'){

for(char\* withS = msg; \*withS != '\0'; withS++){

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

}

}

}

}