6.4 Array/vector iteration drill

Students:
Section 6.5 is a part of 2 assignments: CSC108 CH06.1-6.9 C6A

Please browse to this assignment through Blackboard,learn so zybooks knows where to send your activity. Learn more

Includes: CA
Due: 04/29/2025, 11:59 PM EDT

6.5 Iterating through vectors

Iterating through vectors using loops

Iterating through vectors using loops is commonplace and is an important programming skill to master. Because vector indices are numbered 0 to N - 1 rather than 1 to N, programmers commonly use this for loop structure:

Figure 6.5.1: Common for loop structure for iterating through a vector.

```
// Iterating through myVector
for (i = 0; i < myVector.size(); ++i) {
  // Loop body accessing myVector.at(i)
}
```

Feedback?

Note that index variable i is initialized to 0, and the loop expression is i < myVector.size() rather than i <= myVector.size(). If myVector.size() were 5, the loop's iterations would set i to 0, 1, 2, 3, and 4, for a total of 5 iterations. The benefit of the loop structure is that each vector element is accessed as myVector.at(i) rather than the more complex myVector.at(i - 1).

PARTICIPATION ACTIVITY 6.5.1: Iterating through a vector.

1) Complete the code to print all items for the given vector, using the above common loop structure.

```
vector<int> daysList(365);

for (i = 0;                    ; ++i) {
  cout << daysList.at(i) << endl;
}
```

Check Show answer

2) Given that this loop iterates over all items of the vector, how many items are in the vector?

```
for (i = 5; i < 95; ++i) {
  cout << someVector.at(i) << endl;
}
```

Check Show answer

Feedback?

Determining a quantity about a vector's items

Iterating through a vector for various purposes is an important programming skill to master. Programs commonly iterate through vectors to determine some quantity about the vector's items. The example below computes the sum of a vector's element values.

Figure 6.5.2: Iterating through a vector example: Program that finds the sum of a vector's elements.

```
#include <iostream>
#include <vector>
using namespace std;

int main() {
  const int NUM_ELEMENTS = 5;     // Number of elements to add
  vector<int> userVals;           // Empty vector for user values
  unsigned int i;                 // Loop index
  int val;                        // Element value
  int sumVal;                     // Sum of elements

  cout << "Enter " << NUM_ELEMENTS << " integer values..." << endl;
  for (i = 0; i < NUM_ELEMENTS; ++i) {
    cin >> val;  /* Read next element */
    userVals.push_back(val);
  }

  sumVal = 0;
  for (i = 0; i < userVals.size(); ++i) {
    sumVal += userVals.at(i);
  }

  cout << "Sum of elements: " << sumVal << endl;

  return 0;
}
```

```
Enter 5 integer values...
Enter value: 7
Enter value: 1
Enter value: 3
Enter value: -4
Enter value: 128
Enter value: 14
Enter value: 26
Enter value: -9
Sum of elements: 237
```

Feedback?

Finding the maximum value in a vector

The program below determines the maximum value in a user-entered list. If the user enters numbers 7, -9, 55, 44, 20, -400, 0, 2, then the program will output "max: 55". The program uses the variable maxVal to store the largest value seen thus far as the program iterates through the vector. Initially, the vector's current element value is larger than the max seen thus far, the program assigns maxVal with the current vector element.

Before entering the loop, maxVal must be initialized to some value because maxVal will be compared with each vector element's value. A logical error would be to initialize maxVal to 0, because 0 is not in fact the largest value seen so far, and would result in incorrect output (of 0) if the user entered all negative numbers. Instead, the program peeks at a vector element (in this case the first element, through any element could be used) and initializes maxVal with that element's value.

Figure 6.5.3: Iterating through a vector example: Program that finds the max item.

```
#include <iostream>
#include <vector>
using namespace std;

int main() {
  const int NUM_ELEMENTS = 5;     // Number of elements to add
  vector<int> userVals;           // Empty vector for user values
  unsigned int i;                 // Loop index
  int val;                        // Element value
  int maxVal;                     // Max of elements

  cout << "Enter " << NUM_ELEMENTS << " integer values..." << endl;
  for (i = 0; i < NUM_ELEMENTS; ++i) {
    cin >> val;  /* Read next element */
    userVals.push_back(val);
  }

  maxVal = userVals.at(0);
  for (i = 0; i < userVals.size(); ++i) {
    if (userVals.at(i) > maxVal) {
      maxVal = userVals.at(i);
    }
  }

  cout << "Max of elements: " << maxVal << endl;

  return 0;
}
```

```
Enter 5 integer values...
Enter value: 23
Enter value: 1
Enter value: 27
Enter value: 93
Enter value: 14
Enter value: -5
Enter value: 35
Max of elements: 107
```

Feedback?

PARTICIPATION ACTIVITY 6.5.2: Iterating through vectors.

Complete the code provided to achieve the desired goal.

1) Find the minimum element value in vector `valVctr`.

```
temp(s) = valVctr.at(0);
for (i = 0; i < valVctr.size(); ++i) {
  if (valVctr.at(i)                    ) {
    temp(s) = valVctr.at(i);
  }
}
```

Check Show answer

2) Find the sum of all elements in vector `valVctr`.

```
valSum =

for (i = 0; i < valVctr.size(); ++i) {
  valSum += valVctr.at(i);
}
```

Check Show answer

3) Count the number of negative-valued elements in vector `valVctr`.

```
numNeg = 0;
for (i = 0; i < valVctr.size(); ++i) {
  if (valVctr.at(i) < 0) {
    numNeg =                    ;
  }
}
```

Check Show answer

Feedback?

zyDE 6.5.1: Computing the average of a vector's element values.

Complete the code to compute the average of the vector's element values. The result should be 16.

Load default template... Run

```
#include <iostream>
#include <vector>
using namespace std;

int main() {
  vector<int> valVctr;      // Empty vector for v
  unsigned int i;           // Loop index
  int sumVal;               // Sum of values
  int avgVal;               // Average of values

  valVctr.push_back(30);
  valVctr.push_back(20);
  valVctr.push_back(10);
  valVctr.push_back(25);
  valVctr.push_back(15);
  valVctr.push_back(10);

  for( i = 0; i < valVctr.size(); ++i){
    cout << "valVctr.at(" << i << ") = " << valVc
```

```
30
20
10
25
15
10
```

Feedback?

Common error: Accessing out of range vector element

A common error is to try to access a vector with an index that is out of the vector's index range. Ex: Trying to access highScores.at(8) when highScores valid indices are 0-7. Care should be taken whenever a user enters a number that is then used as a vector index, and when using a loop index as a vector index also, to ensure the array index is within a vector's valid index range. Accessing an index that is out of range causes the program to automatically abort execution, typically with an error message being automatically printed. For example, for the declaration `vector<int> highScores(8);`, accessing highScores.at(8), or highScores.at(i) where i is 8, yields the following error message when running the program compiled with g++:

Figure 6.5.4: Sample error message when accessing an out-of-range vector index.

```
terminate called after throwing an instance of 'std::out_of_range'
  what():  vector::_M_range_check
Abort
```

Feedback?

zyDE 6.5.2: Loop expressions.

Run the program, which prints the contents of the vals vector. Modify the program's loop expression to be `i <= valVctr.size()` rather than `i < valVctr.size()`, and observe that the program aborts.

Load default template... Run

```
#include <iostream>
#include <vector>
using namespace std;

int main() {
  vector<int> valVctr;      // Empty vector for v
  unsigned int i;           // Loop index

  valVctr.push_back(30);
  valVctr.push_back(20);
  valVctr.push_back(10);
  valVctr.push_back(25);
  valVctr.push_back(15);
  valVctr.push_back(10);

  for( i = 0; i < valVctr.size(); ++i){
    cout << "valVctr.at(" << i << ") = " << valVc
```

```
30
20
10
25
15
10
```

Feedback?

PARTICIPATION ACTIVITY 6.5.3: Iterating through a vector.

Given the following code:

```
const int NUM_ELEMENTS = 3;
vector<int> myVctr(NUM_ELEMENTS);
unsigned int i;
```

1) The normal for loop structure iterates as long as `i <= myVctr.size()`

- True
- False

2) To compute the sum of elements, a reasonable statement preceding the for loop is `int sumVal = 0;`

- True
- False

3) To find the maximum element value, a reasonable statement preceding the for loop is `int maxVal = 0;`

- True
- False

Feedback?

CHALLENGE ACTIVITY 6.5.1: Enter the output for the vector elements.

Challenge activity

Start

Type the program's output

```
#include <iostream>
#include <vector>
using namespace std;

int main() {
  vector<int> userVals;
  unsigned int i;

  userVals.push_back(1);
  userVals.push_back(9);
  userVals.push_back(4);

  for (i = 0; i < userVals.size(); ++i) {
    cout << userVals.at(i) << endl;
  }

  return 0;
}
```

```
1
9
4
```

1 2 3 4 5

Check Next

Feedback?

CHALLENGE ACTIVITY 6.5.2: Iterating through vectors.

Challenge activity

Start

Integer vecLength is read from input as the number of elements in vector yearlyMiles. Given the integer vector yearlyMiles, write a for loop to output the integers in yearlyMiles in reverse order. Output each integer followed by a vertical bar surrounded by spaces (" | ").

Ex: If the input is:

```
4
114 105 114 123
```

then the output is:

```
123 | 114 | 105 | 104 |
```

```
#include <iostream>
#include <vector>
using namespace std;

int main() {
  vector<int> yearlyMiles;
  int vecLength;
  int i;
  int mile;

  cin >> vecLength;

  for (i = 0; i < vecLength; ++i) {
    cin >> mile;
    yearlyMiles.push_back(mile);
  }

  /* Your code goes here */
```

1 2 3

Check Next level

Feedback?

How was this section? Provide section feedback

Activity summary for assignment: CSC108 CH06.1-6.9 C6A

Due: 04/29/2025, 11:59 PM EDT

0 / 40 points

Please browse to this assignment through Blackboard,learn so zybooks knows where to send your activity. Learn more

Completion details