Midterm of CS 135, v2
Oct 17, 2024

Name:	
EmpID:	

1 (30 points) Answer the following questions.

(1) Given `char arr[] = {'A', 'B', 'C'}`, what is arr[2]?

Answer: arr[2] is `'C'`, the third item of array `arr`.

(2) Declare function **decrease**, given an integer array `arr` with `size` many elements, decrease each element of the array by 1. Return type is void. Define the function header (no implementation is needed).

Answer: `void decrease(int arr[], int size);` or `void decrease(int* arr, int size);`

Warning: `void decrease(int& arr[], int size);` is wrong, need to replace `int& arr[]` by `int arr[]`.

(3) Assume that n is properly declared and initialized. Write a statement to declare `lastTwoDigits` as an integer and initialize it to be the two least significant digits of integer n . Suppose n is 123, after the statement, `lastTwoDigits` is 23.

Answer: `int lastTwoDigits = n % 100;`

(4) What is the output?

```
string tens_name(int n);

int main() {
    cout << tens_name(93) << endl;
    return 0;
}

string tens_name(int n) {
    if (n < 20 || n > 99)
       return "";

    string names[] = {"", "", "twenty", "thirty", "forty",
        "fifty", "sixty", "seventy", "eighty", "ninety"};

    return names[n / 10];
}
```

Answer: ninety

(5) Given `string greeting = "Nice to meet you.";` What is the value for greeting.substr(2, 5)?

Answer: substring "ce to"

(6) What is the value of `1 + 5 % 2`?

Answer: 2

(7) The volume of a sphere with radius r is $4/3\pi r^3$. Assume that r is properly declared as double type and initialized, write a statement to declare *volume* and save the value of the volume. Use `M_PI` for π.

Answer: `double volume = 4 / 3.0 * M_PI * r * r * r;` or

`double volume = 4 / 3.0 * M_PI * pow(r, 3);`

(8) What is the output of the following code?

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

int main() {
    int count = 0;
    for (int i = 3; i < 10; i += 2)
        count++;

    cout << count << endl;
    return 0;
}
```

Answer: 4

(9) Write a statement to call `foo` function on double variables a and b, both are properly declared and initialized.

```
void foo(double& a, double& b);
```

Answer: foo(a, b);

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

void foo(double& a, double& b);

int main() {
    double a = 3.1;
    double b = 5.6;
    foo(a, b);
```

```
    cout << "a = " << a << ", b = " << b << endl;
    return 0;
}

void foo(double& a, double& b) {
   double temp = a;
   a = b;
   b = temp;
}
```

(10) Write a condition in C++ to represent that an integer variable n is out of the range of [60, 80], that is, n is either smaller than 60 or larger than 80.

Answer: solution 1: `n < 60 || n > 80`

solution 2: `n < 60 or n > 80`

2 (20 points) Answer the following questions.

(1) What is the output of foo(4, 5)?

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

void foo(int width, int height) {
    int mid;
    if (height % 2 != 0)
       mid = height / 2;
    else mid = height / 2 - 1;

    for (int row = 0; row < height; row++) {
        for (int col = 0; col < width; col++) {
            if (height % 2 != 0) {
               if (row == mid)
                  cout << "*";
               else cout << "-";
            }
            else //now height % 2 == 0
            {
               if (row == mid || row == mid + 1)
                  cout << "*";
               else cout << "-";
            }
        }

        cout << endl;
```

```
    }
}
```

Answer:

```
----
----
****
----
----
```

(2) Define function `numLowercase`, for a string, return the number of characters that are lowercase letters, that is, letter from 'a' to 'z'. **No need to include libraries.**

Hint: you may use `int islower(int ch)` to test whether a character is lowercase letter or not.

Define main function with the following requests.

- Enter two strings from console. The strings may contain spaces.
- If both strings have the same number of lowercase letters, report "the strings have the same number of lowercase letters.", otherwise, find out and print the string with more lowercase letters. Some sample outputs are as follows.

```
Enter the first string: abcd A
Enter the second string: bcd BB
abcd A has more lowercase letters

Enter the first string: bcAB
Enter the second string: cd CD
the strings have the same number of lowercase letters
```

Answer:

```
//Sample input/output:
//Enter the first string: abcd A
//Enter the second string: bcd BB
//abcd A has more lowercase letters

//Enter the first string: bcAB
//Enter the second string: cd CD
//the strings have the same number of lowercase letters

#include <iostream>
using namespace std;

int numLowercase(string str);
```

```
14
15 int main() {
16     cout << "Enter the first string: ";
17     string s1;
18     getline(cin, s1);
19
20     cout << "Enter the second string: ";
21     string s2;
22     getline(cin, s2);
23
24     int n1 = numLowercase(s1);
25     int n2 = numLowercase(s2);
26
27     if (n1 > n2)
28        cout << s1 << " has more lowercase letters" << endl;
29     else if (n2 > n1)
30            cout << s2 << " has more lowercase letters" << endl;
31     else cout << "the strings have the same number of lowercase letters" << endl;
32
33     return 0;
34 }
35
36 int numLowercase(string str) {
37    int count = 0;
38    for (int i = 0; i < str.length(); i++)
39        if (str[i] >= 'a' && str[i] <= 'z')
40           count++;
41
42    return count;
43 }
```

3 (50 points) Programming exercises

(1) Define function called `sumOddFactors`, for a positive integer, sum up its $non-trivial$ factors that are odd. A non-trivial factor of n is a factor of n other than 1 and itself.

Hint: $n/2$ may be a non-trivial factor of n.

Answer:

On implementation is as follows.

```
1 #include <iostream>
2 using namespace std;
3
4 int sumOddFactors(int n);
```

```
5
6  int main() {
7      cout << sumOddFactors(6) << endl; //3
8
9      cout << sumOddFactors(15) << endl; //8
10     return 0;
11 }
12
13 int sumOddFactors(int n) {
14    int sum = 0;
15
16    //possible non-trivial odd factor starts from 3
17    for (int i = 3; i <= n/2; i += 2) {
18        if (n % i == 0) //i is a factor of n
19           sum += i;
20    }
21
22    return sum;
23 }
```

Another implementation is as follows.

```
1  #include <iostream>
2  using namespace std;
3
4  int sumOddFactors(int n);
5
6  int main() {
7      cout << sumOddFactors(6) << endl; //3
8
9      cout << sumOddFactors(15) << endl; //8
10     return 0;
11 }
12
13 int sumOddFactors(int n) {
14    int sum = 0;
15
16    for (int i = 2; i <= n/2; i++) {
17        //Approach 1:
18        //if (n % i == 0)
19        //   if (i % 2 != 0)
20        //      sum += i;
21
22        //Approach 2:
23        if (n % i == 0 && i % 2 != 0)
24           sum += i;
25    }
```

```
26
27      return sum;
28  }
```

In main function, call function on integer 15. Print out the return. **Just write the statements in main function, no need to include libraries**.

Answer:

```
1  cout << sumOddFactors(15) << endl; //8
```

(2) Write code in main to enter a full name in the format "LastName,FirstName" (without quotes), extract the first name and last name and get the initial with the first letter of firstName followed by the first letter of last name. **No need to include libraries**.

Here is a sample input/output, input is highlighted:

```
1  Enter full name in the format of lastName,firstName: Washington,George
2  Initial for Washington,George is GW
```

Hints:

- Find out the index of the character separating first name and last name.

  `size_t find (char c, size_t pos = 0) const;`

  Searches the string for the first occurrence of character `c`. If you do not specify parameter `pos`, then the search starts from the beginning of the string. `size_t` is non-negative integer.
- Extract last name and first name.

  `string substr (size_t pos = 0, size_t len = npos) const;`

  Generate substring that is the portion of the object that starts at character position `pos` and spans `len` characters (or until the end of the string, whichever comes first). If the second parameter `len` is not provided, return a substring starting from `pos` all the way to last character.
- Initialize the result to be an empty string.
- Use concatenate operator + to add the first letter of first name to the result.
- Use concatenate operator + to add the first letter of last name to the result.

Answer:

```
1  #include <iostream>
2  #include <string>
3  using namespace std;
4
5  int main() {
6      cout << "Enter full name in the format of lastName,firstName: ";
7      string fullName;
8      cin >> fullName; //getline(cin, fullName); is also fine
```

```
    int commaIdx = fullName.find(',');
    string lastName = fullName.substr(0, commaIdx - 1);
    string firstName = fullName.substr(commaIdx + 1);

    //warning: cannot write string initial = firstName[0] + lastName[0]; or
    //string initial = "" + firstName[0] + lastName[0];
    string initial = "";
    initial += firstName[0];
    initial += lastName[0];
    cout << "Initial for " << fullName << " is " << initial << endl;

    return 0;
}
```

(3) Define function `shorterThan`, given an array of strings, and its size, together with a target string, if every element of the array has fewer characters than that of the target, then return true. If there is at least one element of the array has same number as or more number of characters than that of the target, return false.

Hint: `length` or `size` method of string class returns the number of characters of that string.

Answer:

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

bool shorterThan(string arr[], int size, string target);

int main() {
    string arr[] = {"abc", "bcef", "bcb"};
    int size = sizeof(arr) / sizeof(arr[0]);

    cout << boolalpha << shorterThan(arr, size, "abcde") << endl; //true
    cout << boolalpha << shorterThan(arr, size, "abc") << endl; //false

    return 0;
}

bool shorterThan(string arr[], int size, string target) {
    for (int i = 0; i < size; i++)
        if (arr[i].length() >= target.length())
          //length can be replaced by size
          return false;

    return true;
```

```
}
```

In main function, write the following statements.

- Declare an array of strings, call it `arr`, initialized with elements `"abc", "bcef"`.
- Call the above function on `arr` and target `"abcde"`, print out the result.

Answer:

```
string arr[] = {"abc", "bcef"};
int size = sizeof(arr) / sizeof(arr[0]);

cout << boolalpha << shorterThan(arr, size, "abcde") << endl; //true
```