Midterm of CS 135, v2
Mar 18, 2024

Name:	
EmpID:	

1 (30 points) Answer the following questions.

(1) Given `string arr[] = {"Hello", "Hi", "Hey"}`, what is arr[1]?

Answer: arr[1] is the second element of arr – the first element is indexed at 0. That is, arr[1] is `"Hi"`.

(2) Declare function **insert**, for an given array `arr` of integers, an integer called `size`, which represents the number of elements in the array, and an integer `capacity`, which represents the capacity of the array, and an integer called `element`. If `size` is smaller than `capacity`, insert `element` to the end of the array and increase the original size by 1, otherwise, do nothing. Return type is void. **Just write function header.**

Answer: `void insert(int arr[], int& size, int capacity, int element)`.

Note that & after int in declaration of size means this parameter is passed by reference. Value of `size` might be changed in function `insert` and its change should be carried back to a caller of `insert` function.

(3) Write a statement to generate a random int in [-10, 50]. No need to include libraries.

Answer: `rand() % 61 -10`

Explanation

(a) There are (50 - (-10) + 1) = 61 integers in range [-10, 50].

(b) rand() % 61 generates a random integer in [0, 60].

(c) rand() % 61 - 10 generates a random integer in [-10, 50].

(4) Given `string greeting = "memory";` What is the value for greeting.substr(2, 3)?

Answer: mor

Explantion: `greeting.substr(2, 3)` returns a substring of `greeting` starting from index 2 and take 3 more characters. The character indexed at 2 of `"memory"` is the second appearance of `m`, and take three more characters from there (including `m` itself), the result is substring `"mor"`

(5) What is the value of 1.0 / 2 * 5?

Answer: 2.5

Explanation: division operator / and multiplication operator * has same precedence and they run from left to right. So 1.0 / 2 runs first, which returns 0.5. Then 0.5 * 5 returns 2.5.

(6) Suppose n is 135, what the value of foo(n)?

```
int foo(int n) {
   int result = 1;
   do {
```

```
        result *= n % 10;
        n /= 10;
    } while (n != 0);

    return result;
}
```

Answer: 15

Key parts:

- n % 10 returns the last digit, aka least-significant digit, of n.
- n = n / 10; or n /= 10; removes the last digit from n.

variable	value
n	135 (formal parameter n is initialized when function foo is called)
result	1

(a) `result *= n % 10;` is the same as `result = result * (n % 10);`, where the original value of `result` is 1 and `n % 10` is the last digit of `n`. When `n` is 135, `n % 10` is like to divide 135 pens among 10 students, each student get 13 pens, and there are 5 pens left, here % is remainder operator.

(b) `n /= 10;` is the same as `n = n / 10;` . When `n` is 135, `n / 10` returns 13.

(c) The exit check condition in do-while statement is `n != 0`. When `n` is 13, the condition holds. So we stay inside the loop.

(d) Run statement `result *= n % 10;`

i. The current value of `result` is 5.

ii. The current value of `n` is 13. Then `n % 10` returns 3.

iii. `result *= n % 10;` is the same as `result = result * (n % 10);`, so after the statement, `result` is changed to be `5 * 3 = 15`.

(e) Run statement `n /= 10;`, which is the same as `n = n / 10;` . The original value of `n` is 13, after `n / 10`, we get 1, which is put in `n`. That is, n is changed to be 1.

(f) Since n is 1 and satisfies `n != 0` in exit check condition of the do-loop. Go back to the loop.

(g) Run statement `result *= n % 10;`

i. The current value of `result` is 15.

ii. The current value of `n` is 1. Then `n % 10` returns 1.

iii. `result *= n % 10;` is the same as `result = result * (n % 10);`, so after the statement, `result` is changed to be `15 * 1 = 15`.

(h) Run statement `n /= 10;`, which is the same as `n = n / 10;` . The current value of `n` is 1, after `n / 10`, we get 0, which is put in `n`. That is, n is changed to be 0.

(i) The exit check condition in do-while statement is `n != 0`. When `n` is 0, the condition does not holds. So we exit the do-while loop.

(j) Return the value of `result`, which is 15, the value of `foo(135)`.

(7) Given two double variables a and b, find out the return of b^a. Hint: use pow function, see cheat sheet paper.

Answer: pow(b, a);

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

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

int main() {
    int count = 0;
    for (int i = 2; i >= -1; i -= 2)
        count++;

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

Answer: 2

(9) What is the output for the following code?

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

void foo(int& a, int b);

int main() {
    int i = 2;
    int j = 6;

    foo(i, j);

    cout << "i = " << i << ", j = " << j << endl;
    return 0;
}

void foo(int& a, int b) {
    a++;
    b*=2;
}
```

Answer: i = 3, j = 6

(10) Write a condition to represent that char variable *ch* is none of the following: '1', '2', or '3'.

Answer: `(ch != '1' && ch != '2' && ch != '3')` or `! (ch == '1' || ch == '2' || ch == '3')`

2 (20 points) Answer the following questions.

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

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

void show(int height, int width);

int main() {
    show(4, 6);
    return 0;
}

void show(int height, int width) {
    for (int row = 0; row < height; row++) {
        for (int col = 0; col < width; col++)
            if (row >= height/2 && col >= width/2)
               cout << "*";
            else cout << "-";

        cout << endl;
    }
}
```

Answer:

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

(2) Define function `all_small_letters`, for an array of chars, return true if **all** its elements are smaller letters ('a' to 'z'), otherwise, return false. **No need to include libraries or define main function.**

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

Answer:

```
#include <iostream>
#include <cctype> //islower function
#include <string>
using namespace std;

bool all_small_letters(char arr[], int size);

int main() {
    char arr[] = {'a', 'b', 'c'};
    int size = sizeof(arr) / sizeof(arr[0]);
```

```
    cout << boolalpha << all_small_letters(arr, size) << endl;

    char arr2[] = {'a', 'b', 'c', 'D'};
    int size2 = sizeof(arr2) / sizeof(arr2[0]);
    cout << boolalpha << all_small_letters(arr2, size2) << endl;

    return 0;
}

bool all_small_letters(char arr[], int size) {
    for (int i = 0; i < size; i++)
        //if ( !(arr[i] >= 'a' && arr[i] <= 'z') ) //ok
        if ( !islower(arr[i]) ) //also ok
           return false;

    return true;
}
```

3 (50 points) Programming exercises

(1) Define function called `tally`, for an array of integers, calculate the number of positive integers, negative integers, and zeros, then return the mininum of these three numbers. For example, if the array has elements 1, 2, -1, -2, 0, the return is 1.

Answer:

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

int tally(int arr[], int size);

int main() {
    int arr[] = {1, 2, -1, -2, 0};
    int size = sizeof(arr) / sizeof(arr[0]);
    cout << tally(arr, size) << endl;
    return 0;
}

int tally(int arr[], int size) {
   int numPos = 0;
   int numNeg = 0;
   int numZeros = 0;
   for (int i = 0; i < size; i++)
       if (arr[i] > 0)
          numPos++;
       else if (arr[i] < 0)
              numNeg++;
           else numZeros++;

   int min = numPos;
   if (numNeg < min)
      min = numNeg;
   if (numZeros < min)
      min = numZeros;

   return min;
}
```

In main function, declare array `arr` with values 1, 2, -1, -2, 0, and call the above function on `arr`. Print out the return. **Just write the statements in main function, no need to include libraries**.

Answer:

```
    int arr[] = {1, 2, -1, -2, 0};
    int size = sizeof(arr) / sizeof(arr[0]);
    cout << tally(arr, size) << endl;
```

(2) Write code in main to enter strings (may contain spaces) until user enter "exit". Tally how many string has odd number of characters.

Here is a sample input/output. `Enter a series of strings until enter "exit"` is a prompt. String `try` has odd length.

```
Enter a series of strings until enter "exit"
No, no
try
ABC DE
exit
number of odd strings: 1
```

Answer:

```
//sample input/output:
//Enter a series of strings until enter "exit"
//No, no
//try
//ABC DE
//exit
//number of odd strings: 1

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

int main() {
    cout << "Enter a series of strings until enter \"exit\"" << endl;
    string str;
    getline(cin, str);

    int numOddStrs = 0;
    while (str != "exit") {
        if (str.size() % 2 != 0)
            numOddStrs++;

        getline(cin, str);
    }

    cout << "number of odd strings: " << numOddStrs << endl;
}
```