

C and C++ Programming Assessment 3 Answers

August 25, 2025

C MULTIPLE CHOICE ANSWERS

1. **B) 8 bytes**

A `double` typically occupies 8 bytes on most systems, following the IEEE 754 double-precision format.

2. **C) string**

C does not have a built-in `string` data type; strings are handled as arrays of `char`. Other options are valid C types.

3. **D) Undefined Behavior**

The expression `++x + x++` modifies `x` multiple times without a sequence point, leading to undefined behavior.

4. **A) Pointer not initialized**

The pointer `ptr` is not initialized before dereferencing, causing undefined behavior when assigning `*ptr = 10`.

5. **D) Both B and C**

Both `while` and `for` loops check their condition before executing the body, unlike `do-while`.

6. **C) Size of a pointer**

When an array is passed to a function, it decays to a pointer, so `sizeof` returns the size of the pointer (typically 4 or 8 bytes).

7. **A) Loop condition is incorrect**

The loop condition `i <= 5` accesses `arr[5]`, which is out of bounds for `arr[5]` (indices 0 to 4).

8. **A) Defines a variable as immutable**

The `const` keyword declares a variable as read-only, preventing modification after initialization.

9. **B) `int arr[3][3];`**

A 2D array is declared as `int arr[rows][cols]`. Option A uses incorrect syntax, C is invalid, and D declares an array of pointers.

10. **A) Missing variable in `fscanf`**

`fscanf` requires a variable to store the scanned value (e.g., `fscanf(fp, "%d", &x)`), otherwise it causes undefined behavior.

C++ MULTIPLE CHOICE ANSWERS

1. **B) References**

References are a C++ feature, not available in C. Pointers, arrays, and structures exist in both languages.

2. **B) Missing semicolon after class**

A class definition requires a semicolon after the closing brace. The syntax for `public` is correct but lacks the semicolon.

3. **B) Function overrides a base class virtual function**

The `override` keyword ensures a function overrides a virtual function in the base class, preventing errors.

4. B) Constructors cannot return values

Constructors initialize objects and cannot return values, even local variables like `x`.

5. B) Current object instance

The `this` pointer refers to the current object instance within a class's member functions.

6. A) Incorrect deletion syntax for single object

A single object allocated with `new` should be deleted with `delete`, not `delete[]`, which is for arrays.

7. A) To redefine operators for user-defined types

Operator overloading allows custom behavior for operators (e.g., `+`) with user-defined classes.

8. B) final

The `final` keyword prevents a class from being inherited or a virtual function from being overridden.

9. A) Default access specifier

In a `struct`, members are `public` by default; in a `class`, they are `private`.

10. B) Derived

The `virtual` function enables dynamic binding, so the `print` function of `Derived` is called, outputting "Derived".

C CREATIVE QUESTION ANSWERS

1. Array Definition and Printing

```
1      int main() {
2          int arr[8] = {5, 10, 15, 20, 25, 30, 35, 40};
3          for (int i = 0; i < 8; i++) {
4              printf("%d ", arr[i]);
5          }
6          printf("\n");
7          return 0;
8      }
```

Initializes and prints the array to confirm.

2. Reverse Array Function

```
1      void reverseArray(int arr[], int size) {
2          for (int i = 0; i < size / 2; i++) {
3              int temp = arr[i];
4              arr[i] = arr[size - 1 - i];
5              arr[size - 1 - i] = temp;
6          }
7      }
```

Reverses the array in place by swapping elements from the ends.

3. Modify main with Reverse Array

```
1      int main() {
2          int arr[8] = {5, 10, 15, 20, 25, 30, 35, 40};
3          printf("Original array: ");
4          for (int i = 0; i < 8; i++) {
5              printf("%d ", arr[i]);
6          }
```

```

7         printf("\n");
8         reverseArray(arr, 8);
9         printf("Reversed array: ");
10        for (int i = 0; i < 8; i++) {
11            printf("%d ", arr[i]);
12        }
13        printf("\n");
14        return 0;
15    }
16

```

Calls reverseArray and prints the reversed array.

4. Find Duplicates

```

1    void findDuplicates(int arr[], int size) {
2        int found = 0;
3        for (int i = 0; i < size; i++) {
4            for (int j = i + 1; j < size; j++) {
5                if (arr[i] == arr[j]) {
6                    printf("Duplicate found: %d\n", arr[i]);
7                    found = 1;
8                }
9            }
10        }
11        if (!found) printf("No duplicates found\n");
12    }
13

```

Checks for duplicates and prints them; called in main.

5. Calculate Range

```

1    int calculateRange(int arr[], int size) {
2        int max = arr[0], min = arr[0];
3        for (int i = 1; i < size; i++) {
4            if (arr[i] > max) max = arr[i];
5            if (arr[i] < min) min = arr[i];
6        }
7        return max - min;
8    }
9

```

Returns the range (max - min), printed in main.

C++ CREATIVE QUESTION ANSWERS

1. Book Class Definition

```

1    class Book {
2    private:
3        string title;
4        float ratings[4];
5        int isbn;
6    public:
7        Book(string t, float r[], int i) {
8            title = t;
9            for (int j = 0; j < 4; j++) ratings[j] = r[j];
10           isbn = i;
11        }
12    };
13

```

Defines the class with private members and a constructor.

2. Calculate Average Rating

```
1     float calculateAverageRating() {
2         float sum = 0;
3         for (int i = 0; i < 4; i++) {
4             sum += ratings[i];
5         }
6         return sum / 4;
7     }
8
```

Computes and returns the average rating.

3. Is Highly Rated

```
1     bool isHighlyRated() {
2         return calculateAverageRating() >= 4.0;
3     }
4
```

Returns true if the average rating is 4.0 or higher.

4. Display Book Info

```
1     void displayBookInfo() {
2         cout << "Title: " << title << endl;
3         cout << "ISBN: " << isbn << endl;
4         cout << "Ratings: ";
5         for (int i = 0; i < 4; i++) {
6             cout << ratings[i] << " ";
7         }
8         cout << endl;
9         float avg = calculateAverageRating();
10        cout << "Average Rating: " << fixed << setprecision(2) << avg <<
endl;
11        cout << "Status: " << (isHighlyRated() ? "Highly Rated" : "Not
Highly Rated") << endl;
12    }
13
```

Displays book details in the specified format.

5. Main Function

```
1     int main() {
2         float ratings1[4] = {4.5, 4.0, 3.8, 4.2};
3         float ratings2[4] = {3.5, 3.0, 3.2, 3.8};
4         Book book1("The Great Gatsby", ratings1, 123456789);
5         Book book2("1984", ratings2, 987654321);
6         book1.displayBookInfo();
7         cout << endl;
8         book2.displayBookInfo();
9         return 0;
10    }
11
```

Creates two Book objects and tests all functionalities.