OOP@2024 Quiz 2 **8 May 2024**

1. Which of the following descriptions about program output is correct?

```cpp
#include <iostream>
#include <vector>

int main() {
    std::vector<int> vec = {10, 20, 30, 40, 50};

    vec.push_back(60);
    vec.insert(vec.begin() + 2, 25);

    vec.erase(vec.begin() + 4);
    vec.pop_back();

    for(int i = 0; i < vec.size(); ++i) {
        std::cout << vec[i] << " ";
    }

    return 0;
}
```

   (a) `10 20 30 40 50 60`

   (b) `10 20 25 30 40 50`

   (c) `10 20 25 30 50`

   (d) `10 20 25 30 40`

2. In C++, which of the following statements about function overloading is correct?

   (a) Function overloading can be defined by the return type

   (b) Function overloading can be defined merely by the difference in the const attribute

   (c) Function overloading can be defined by the differences in parameter type, number, or order

   (d) Definition of function overloading solely based on the differences in default parameters is valid

3. Which of the following descriptions about program output is correct?

```cpp
struct A {
  virtual void f() { cout << "A"; }
  void g() { cout << "a"; }
};
struct B : A {
  void f() { cout << "B"; }
  void g() { cout << "b"; }
};
```

```
A* get_object( bool w ) {
  return w ? new A() : new B();
}
int main() {
  auto p = get_object( true ) , q = get_object( false );
  p -> f() , p -> g() , q -> f() , q -> g();
}
```

(a) `AaBa`

(b) `AaBb`

(c) `aBaA`

(d) `bBaA`

4. When you are designing a class with at least one virtual method, you'd better declare which function of the following with keyword `virtual`?

   (a) constructors

   (b) destructor

   (c) all overridden functions

   (d) all overloaded operators

5. Which of the following allows a type to be a parameter of a method, class, or interface in C++?

   (a) Template

   (b) Hierarchy

   (c) Polymorphism

   (d) Overloading

6. Which of the following descriptions about program output is correct?

```
#include <iostream>
template<class T>
struct A { T x; A() { std::cout << "A()"; } };
struct B: A { ~B() { std::cout << "~B()"; } };
int main() { B b; }
```

   (a) The program is guaranteed to output `A()~B()`

   (b) The program is guaranteed to output `~B()`

   (c) The program has a compilation error.

   (d) The program is undefined.

7. Which of the following descriptions about program output is correct?

```cpp
#include <iostream>
template <typename T>
T my_max(T x, T y) { return (x > y)? x : y; }
int main() {
    std::cout << my_max(3, 7) << " ";
    std::cout << my_max(3.0, 7.0) << " ";
    std::cout << my_max(3, 7.0) << std::endl;
    return 0;
}
```

(a) The program is guaranteed to output `7 7.0 7.0`

(b) The program has a compilation error because the data type is not specified in all `cout` statements

(c) The program has a compilation error because the call to last call to `max` is ambiguous.

(d) The program is undefined.

8. When is the value calculated in following program?

```cpp
#include <iostream>
template<int N>
struct Fact { static constexpr int value = N * Fact<N - 1>::value; };

template<>
struct Fact<0> { static constexpr int value = 0; };

int main() {
    std::cout << Fact<10>::value;
}
```

(a) When the program is running.

(b) During the compile time.

(c) Before the compiling.

(d) Calculated by author.

9. Explain why `static` member functions cannot be marked `const`.

10. Implement a class and overload the `~` operator in the class (you only need to provide the **core code**, don't consider anything else).

11. **In Task 5, you have completed the following issues.**

    Problem Statement

    John Smith knows that his son, Thomas Smith, is among the best students in his class and even in his school. After the students of the school took

the exams in English, German, Math, and History, a table of results was formed.

There are n students, each of them has a unique id (from 1 to n). Thomas's id is 1. Every student has four scores correspond to his or her English, German, Math, and History scores. The students are given in order of increasing of their ids.

In the table, the students will be sorted by decreasing the sum of their scores. So, a student with the largest sum will get the first place. If two or more students have the same sum, these students will be sorted by increasing their ids.

Please help John find out the rank of his son.

Input Format

The first line contains a single integer n ($1 \leq n \leq 1000$) — the number of students.

Each of the next n lines contains four integers a_i , b_i , c_i , and d_i ($0 \leq a_i, b_i, c_i, d_i \leq 100$) — the grades of the i -th student on English, German, Math, and History. The id of the i -th student is equal to i .

Output Format

Print the rank of Thomas Smith. Thomas's id is 1.

Sample Input

```
5
100 98 100 100
100 100 100 100
100 100 99 99
90 99 90 100
100 98 60 99
```

Sample Output

```
2
```

Now you don't need to redo this question; we will give you the correct code for this question. However, some parts of the code have been "dug out." Your task is to fill in the blanks in the code.

There are **6 blanks** in total marked Q1 to Q6 in the code and you need to answer them respectively.

```cpp
#include <iostream>
#include <vector>
#include <algorithm>

class Student {
public:
    int id;
    int scores[4];

    // Constructor
    Q1{
        scores[0] = english;
        scores[1] = german;
        scores[2] = math;
        scores[3] = history;
    }

    // Calculate the sum of scores
    int totalScore() const {
        return scores[0] + scores[1] + scores[2] + scores[3];
    }

    // Operator overloading to compare students based on their scores
    bool operator<(const Student &other) const {
      Q2{
        return totalScore() > other.totalScore();
      }
      else{
        return Q3;
      }
    }
};

int main() {
    int n;
    std::cin >> n;

    // Reading students' data
    std::vector<Student> students;
    for (int i = 1; i <= n; ++i) {
        int english, german, math, history;
        std::cin >> english >> german >> math >> history;
        students.push_back(Student(i, english, german, math, history));
    }

    // Sorting students in decreasing order of their total scores
    std::sort(Q4);

    // Finding Thomas's rank
    for (int i = 0; i < n; ++i) {
        if (students[i].id == 1) {
            std::cout << Q5 << "\n";
            Q6;
        }
    }

    return 0;
}
```

12. The following code declares a class called singleton, which means that there

can be **only one** instance of the class at any time.

Fill in the blanks Q1 to Q4 using **keywords** in C++ and **try to explain why there can't be more than one instance of the class**.

```
class singleton {
public:
  singleton( const singleton& ) = Q1;
  singleton &operator = ( const singleton& ) = Q2;
  Q3 singleton* get_instance() {
    if( s_instance == nullptr ) {
      s_instance = new singleton();
    }
    return s_instance;
  }
protected:
  singleton() = Q4;
  static singleton* s_instance;
};
singleton* singleton::s_instance = nullptr;
```

13. Read the following program, answer the result of calling

    `sp -> output(),dp -> output(),sq -> output(),dq -> output()`

    separately and explain why.

```
#include <iostream>
using std::cout;
struct A {
    int x;
    virtual void output() {
        cout << "A";
    }
};
struct B : A {
    int y;
    virtual void output() {
        cout << "B";
    }
};
A *p , *q;

int main() {
    p = new A();
    q = new B();
    auto sp = static_cast<B*>( p );
    auto dp = dynamic_cast<B*>( p );
    auto sq = static_cast<B*>( q );
    auto dq = dynamic_cast<B*>( q );
}
```

14. Implement a template class `is_same<T, U>`, when type T = U, the only variable `value` in it is true, otherwise is false. Here is an example of it.

```
int main() {
    std::cout << std::boolalpha;
    std::cout << is_same<int, long long>::value << " " << is_same<int, int>::value <<
     std::endl;
```

```
    // print 'false true'
}
```

15. It is possible to implement polymorphism using template class.

```
#include <iostream>
template<class T>
struct Base {
    T* self() { return static_cast<T*>(this); }
    void func() { Q1; }
};
struct Derived1: public Base<Derived1> {
    void func() { std::cout << "Derived1!\n"; }
};
struct Derived2: public Q2 {
    void func() { std::cout << "Derived2!\n"; }
};

template<class T>
void print(/*no const*/ Q3 d) { d.func(); }

int main() {
    Derived1 d1;
    Derived2 d2;
    print(d1);
    print(d2);
}
```

1. Fill in the blanks Q1 to Q3 respectively.

2. Are the classes `Derived1` and `Derived2` derived from same base class? Why?

3. Compare the polymorphism using template class and using virtual function, show the advantages and disadvantages of using template class.