# Final Exam of Programming Language II (CSE, NTOU) ID: _____

09:20 – 12:05, 6 June 2025; Room INS101 Name: _____

**Note:** Cell phones and any smart device or calculator are forbidden.

1. (5%) Please modify the following program so that it can print **32** as the output successfully, (DO NOT MODIFY `main()` function).

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

class ClassA {
public:
    ClassA() : numA(12) {}
private:
    int numA;
    friend int add(ClassA, ClassB);
};

class ClassB {
public:
    ClassB() : numB(20) {}
private:
    int numB;
    friend int add(ClassA, ClassB);
};

int add(ClassA objectA, ClassB objectB) { return (objectA.numA + objectB.numB);}

int main() { // DO NOT MODIFY IT
    ClassA objectA;
    ClassB objectB;
    cout << "Sum: " << add(objectA, objectB);
    return 0;
}
```

2. (5%) Can a class have a private constructor? (Just answer "*Yes*" or "*No*").

3. (10%) Suppose that we have a class `Vect` for two-dimensional vectors in the following. Please correct the operator overloading of '=' to make the main function smoothly executable.

```
class Vect {
    int x, y;
public:
    Vect(int a, int b): x(a), y(b) {}
    void operator=(Vect r); //correct and complete it
    void print() { cout << x << ", " << y << endl; }
};
```

```
int main() {
    Vect o1, o2;
    Vect o3(1,2);
    o1 = o2 = o3;
    o1.print();
    return 0;
}
```

```
Sample output:
1, 2
```

4. (5%) What is the output of this program? Just write it down.

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

template <class T>
class CHECK {
    public:
      void f() { cout << "CHECK IT OUT!"<< endl ;}
};
template <>
class CHECK<char> {
    public:
      void f() { cout << "CHECK CHAR OUT!"<< endl ;}
};
```

```
int main() {
    CHECK<char> c1;
    CHECK<int> c2;

    c1.f();
    c2.f();
    return 0;
}
```

5. (10%) Please modify class `tList` to be a template so that the main function can run successfully.

```
class tList { // make it a template
     int * head;

public:
     int length;
     tList(int len) {
       head = new int[len];
       length = len;
     }
     ~tList() {
       delete[] head;
     }
     int get(int index) {
       return head[index];
     }
     void set(int index, int val) {
       head[index] = val;
     }
};
```

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

int main() {
    tList<double> nums(5);
    for (int i=0; i<nums.length; i++) {
        nums.set(i, i*1.5);
        cout << nums.get(i) << " ";
    }
    cout << endl;
    return 0;
}
```

Sample output:

```
0 1.5 3 4.5 6
```

6. (5%) Add a lambda expression or any inlinable function to sort an integer vector in the decreasing order (遞減).

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

using namespace std;
// add your code here if needed

int main()
{
    vector<int> v{21,1,13,18,72};
    sort(v.begin(), v.end(), INSERT_YOUR_CODE_HERE);
    cout << v[0] << ", " << v[4] << endl;
    return 0;
}
```

**Hint:** Lambda expression:

```
[capture clause] (parameters){
   definition of method
}
```

Sample output:

```
72, 1
```

7. (10%) Please modify the code below to satisfy the need of the main function.

```
class Base { // make it a template
public:
    Base(double data): mData(data) { }
    virtual void print() { cout << mData << endl; }
protected:
    double mData;
};

class Derived : public Base { // make it a template (derived from Base)
public:
    Derived() = default;
    Derived(double data) : Base(data) { }
    void print() override { cout << "data = " << this->mData << endl; }
};

int main() {
    Derived<int> d1(5);
    Derived<std::string> d2("NTOU_CSE");
    d1.print();
    d2.print();
    return 0;
}
```

8. (5%) Which container is used as a *hash-bucket structure*? `map`, `unordered_map`, or `priority_queue`?

9. (10%) Please **complete the class `date`** and **overload the operator "<<"** to print the `date` object.

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

class date {
private:
    int year, month, day;
public:
    date() = default;
    // complete the constructors
    ~date() = default;
    //overload << using friend function declaration
};
int main() { // DO NOT MODIFY IT
    int y, m, d;
    cin >> y >> m >> d; // sample input: 1998 7 8
    date obj(y, m, d); // sample output: 1998/07/08
    cout << obj << endl;
    return 0;
}
```

10. (5%) Please indicate which one of the following is wrong.

    a. `shared_ptr<int> p2(new int(42));`

    b. `shared_ptr<int> p1 = new int(42);`

11. (5%) Please **correct the following code** which has memory leaks.

```
void leaky() {
    int *p = new int(99);
    std::cout << "value= " << *p << endl;
}
```

12. (10%) Please correct the following code which has the issue of shallow copy.

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

struct point {
    int x, y;
};

class Shallow {
public:
    point* data;
    Shallow(): data(new point(0,0)) {}
    Shallow(point v) { data = new point(v); }
    ~Shallow() { delete data; }
    // Hint: add a copy constructor and overload operator=
};
int main() {
    Shallow a(5), c;
    Shallow b = a;          // shallow copy here
    c = a;                  // shallow copy here
    return 0;
}
```

13. (5%) Please predict the output of the following program.

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

class Base {
   protected:
      int x;
   public:
      Base() { x = 20; }
     friend void display();
};
class Derived : public Base {
   private:
      int y;
   public:
      Derived() { x = 40; }
};
void display() {
   Derived d;
   cout << d.Base::x << endl;
}
int main() {
   display();
   return 0;
}
```

14. (10%) Predict the output of the following program.

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

struct Player {
    std::string name;
    double _fg_percent;
    double _3p_percent;
    double _ft_percent;
    bool operator<(const Player &other) const {
        return this->_3p_percent < other._3p_percent;
    }
};
int main() {
     std::vector<Player> players = {
        {"Anthony",   44.7, 39.5, 83.7},
        {"Brown",     46.3, 32.4, 76.4},
        {"Curry", 44.8, 39.7, 93.3}
    };
    std::priority_queue<Player> pq;
    for (const auto &s : players) { pq.push(s); }
    for (int rank = 0; rank < 3; ++rank) {
        const auto &s = pq.top();
        cout << s.name << endl;
        pq.pop();
    }
    return 0;
}
```

15. (5%) Which STL container supports the last-in-first-out (LIFO) structure?

16. (10%) What kind of exception will be in the following codes? How to “catch” it?

```
std::vector<int> v = { 1, 2 };
v.at(2) = 5;
```