supervisor signature	

2024.2 Object-Oriented Programming
Final Exam (Dec. 16, 8pm-9:20pm)

StudentID# : (   ) , Name : (   )

*** You should write your answer in English. Otherwise, penalty can be applied.**

1. (30 points) Fill in blanks (a)~(t) with the most appropriate English words.

(1) Properties of abstract class:
- An abstract class contains at least one (a.   ).
- No (b.   ) of an abstract class can be created.
- An abstract class can only be used as (c.   ) class.
- One of the main purposes of abstract class is to delegate (d.   ) responsibility to the derived class.

(2) In STL, a container manages storage space for its elements and provides member functions to access them, either directly or through (e.   ).

(3) UML stands for (f. U____________) (g. M______________) Language.
(same as (f)): UML has become a world (h.   ),
(same as (g)): UML describes a software system at a high level of (i.   ),
Language   : UML expresses an idea.

(4) Just like arrays, C++ STL vectors use (j.   ) storage locations for their elements, which means that their elements can also be accessed using offsets on regular pointers to its elements, and just as efficiently as in arrays. But unlike arrays, their (k.   ) can change dynamically, with their storage being handled automatically by the container.

(5) One of the main advantage of STL iterators is that they make the STL algorithm functions (l.   ) of the type of container used. This significantly reduces the complexity of the library.

(6) (m.   ) is a holder object that stores a collection of other objects with any data types.

(7) A **vector**'s member function (n.   ) returns the size of the storage space currently allocated for the vector. A **vector**'s member function (o.   ) returns theoretical limit on the size of the vector.

(8) (p.   ) is a method of passing arguments to a function. In this method, a (q.   ) of the actual parameter's value is passed to the function. This means that any (r.   ) made to the value of the parameter inside the function do not affect the value of the parameter (i.e. argument) outside the function.

(9) (s.   ) relationship: Say "(t.   )". If it sounds right to your ear, then A can be made as a subclass of B.

2. (12 points) Fill in empty blanks with appropriate English words.
(1) What is the main purpose of using **friend** in C++? What does it do? Explain.
([use less than 12 English words]   )

(2) What is the meaning of "**protected**" access (visibility) modifier? Explain.
([use less than 12 English words]   )

(3) What is "function template instantiation" ? Explain.
([use less than 12 English words]   )

3. (12 points) Consider following C++ program and its program output. Fill in the empty box with appropriate C++ code so that the program can generate the output.

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

class myclass
{
    [                                        ]
};
```

```
int main() {
   int arr[] = {1, 2, 3, 4, 5, 6, 7};
   int Size=7;
   myclass X1(1), X2(2), X3(3);

   transform(arr, arr+Size, arr, X1); // apply +1 to each element
   for (auto v: arr) cout << v << " ";
   cout << "\n";
   transform(arr, arr+Size, arr, X2); // apply +2 to each element
   for (auto v: arr) cout << v << " ";
   cout << "\n";
   cout << X3(100) <<"\n"; // apply +3 to argument value and print

   return 0;
}
```

program output:

```
2 3 4 5 6 7 8
4 5 6 7 8 9 10
103
```

4. (8 points) What are the possible purposes of using '**&**' (reference type) in **`void f(T& arg)`** compared to using the parameter type **`void f(T arg)`**? There are two important purposes (i.e. benefits). Explain with sufficient details.

(purpose 1: [use less than 12 English words])

(purpose 2: [use less than 12 English words])

5. (20 points) C++ code below shows generic stack implementation using template. Fill in empty boxes with appropriate C++ code.

```
[                              ]
class Stack {
   int size;
   int top;
   T *stackPtr;
public:
   Stack(int n) { size=n; top=0; [                              ] }
   ~Stack() { delete[] stackPtr; }
   bool push([                ]);  // return true if push is successful
                                   // return false if the stack is full

   bool pop([                 ]);  // return true if pop is successful
                                   // return false if the stack is empty
   bool isEmpty() { if (top<=0) return true; else return false; }
   bool isFull() { if (top>=size) return true; else return false; }
};
// Insert your code for implementing 'push' member function



// Insert your code for implementing 'pop' member function
```

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

int main()
{
  int x, y, z;
  float xf, yf, zf;
  Stack<int> s1(5);
  Stack<float> s2(5);
  s1.push(5); s1.push(8); s1.push(7);
  s1.pop(x);  s1.pop(y); s1.pop(z);
  s2.push(5.3); s2.push(8.1); s2.push(2.5);
  s2.pop(xf);  s2.pop(yf); s2.pop(zf);
  cout << x <<" "<< y <<" "<< z << endl;
  cout << xf <<" "<< yf <<" "<< zf <<endl;

  return 0;
}
```

program output:

```
7 8 5
2.5 8.1 5.3
```

6. (18 points) What is the output of the following C++ program to the screen? Insert your answer into the empty box below.

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

class B {
public:
  B() { z=-8;  cout << "B(): z=" << z << endl; }
  B(int z_val):z(z_val) { cout << "z=" << z << endl;  z++;}
  virtual int get_val() { --z; return (z-1); };
  int gv2() { --z; return (z-2); }
private :
  int z;
};

class D1 : public B {
public:
  D1() { x=5; cout << "D1(): x=" << x << endl; x++; }
  D1(int x_val): x(x_val) { cout << "x=" << x << endl; x--;}
  virtual int get_val() { x++; return x; };
  int gv2() { x++; return x+1; }
private:
  int x;
};

class D2 : public B {
public:
  D2() { y=6; cout << "D2(): y=" << y << endl; }
  D2(int y_val): y(y_val) { cout << "y=" << y << endl; }
  int get_val() { y--;  return y; }
  virtual int gv2() { y--; return y*y; };
private:
  int y;
};

void myf1(B& f) { cout << "0 : " << f.get_val() << endl ; }
void myf2(B& f) { cout << "1 : " << f.gv2() << endl ; }
void myf3(D1 f) { cout << "9 : " << f.get_val() << endl ; }
void myf4(D1 f) { cout << "10: " << f.gv2() << endl ; }
```

```
int main() {
  B Zero(0);   D1 Two;   D2* d2ptr;
  B* B_ptrArray[2];
  B_ptrArray[0] = &Zero;   B_ptrArray[1] = &Two;
  d2ptr = new D2(3);
  myf1(Two);
  myf2(Two);
  cout << "2 : " << B_ptrArray[0]->get_val() << endl;
  cout << "3 : " << Two.get_val() << endl;
  cout << "4 : " << Two.gv2() << endl;
  cout << "5 : " << B_ptrArray[1]->get_val() << endl;
  cout << "6 : " << B_ptrArray[1]->gv2() << endl;
  cout << "7 : " << d2ptr->gv2() << endl;
  cout << "8 : " << d2ptr->get_val() << endl;
  myf3(Two);
  myf4(Two);
  cout << "11: " << Two.get_val() << " " << Two. gv2() << endl;
  delete d2ptr;
  return 0;
}
```

Output : (INSERT YOUR ANSWER HERE)