1. (6 marks) Java program is given:

```
class Person{
        int x;
        public Object work(){return new Person();}
}

class Human{
        int x;
        public Object work(){ return new Person();}
}

class Worker extends Person {
        int x;
        public Object work() { return new Worker();}
        public void overTime(int h) { x = x+h; }

        public static void main(String[] args){
                Person a = new Human();       // line 1
                Worker b = new Person();      // line 2
                Person c = new Worker();      // line 3
                c.overTime(5);                // line 4
                c.x =5;                       // line 5
                Worker m = c.work();          // line 6
        }
}
```

For each line (line 1 to line 6), does it compile? If it does not compile, give the reason.

2. For the code below (a language with nested subroutine), the language uses a value model of variables.

```
program A(){
    x, y, z: integer;
    procedure B(){
        y: integer;
        y=0;
        x=z+1;
        z=y+2;
    }

    procedure C(){
        z: integer;
        procedure D(){
            x: integer;
            x = z + 1;
            y = x + 1;
            call B();
        }
        z = 5;
        call D();
    }
    x = 10;
    y = 11;
    z = 12;
    call C();
    print x, y, z;
}
```

2.1 (3 marks) If the language uses static scoping, the printed result of x, y, and z is	2.2 (3 marks) If the language uses dynamic scoping, the printed result of x, y, and z is
x =	x =
y =	y =
z =	z =

3. Given the C++ code below.

```
class First {
public:
    First() { b = 10; }
    virtual void display(int &x, int y) { x = x + y; cout << "b, x " << b << " " << x << endl; }
private:
    int b;
};

class Second: public First {
public:
    Second() { d = 20; }
    virtual void display(int &x, int y) { x = x * y; cout << "d, x " << d << " " << x << endl; }
private:
    int d;
};

int main() {
    First f, *p;
    Second s;
    int m = 1;
    int *n = new int(2);
    float o = 5.7;
    p = &s;
    p->display(m, o);               //line1
    f = s;
    f.display(m,o);                 //line2
    return 0;
}
```

(1 mark) At line1, the method binding is □ static □ dynamic

(1 mark) At line2, the method binding is □ static □ dynamic

(1 mark) In the checking of the types of the method arguments at line2, the following rule(s) of the type system are used (you may choose 1 or more).

□ type equivalence □ type compatibility □ type inference

4. A Java-like language uses left-to-right evalution order. Its precedence and associativity rules are given below. (Precedence is from the highest downto the lowest.)

Operator	Description	Associativity
…	…	…
* / %	multiplicative	left to right
…	…	…
== !=	equality	left to right
…	…	…
&&	logical and	left to right
||	logical or	left to right
…	…	…

4.1 (3.5 marks) Add parentheses to the expression below to show the effect of precedence and associativity to the grouping of operands to operators.

```
c % 400 == 0 || c % 4 == 0 && c % 100 != 0
```

4.2 (1.5 marks) If c is 1666, the result of the expression is ....

4.3 (3 marks) If this language has short circuiting, which of these subexpressions get evaluated in the question 4.2?

c % 400 == 0	□ yes	□ no
c % 4 == 0	□ yes	□ no
c % 100 != 0	□ yes	□ no