1. What is value of variable **x** after executing each of the following pieces of C++ code

    (a)   (5%)

```
int x = 1;
x = x ++;
```

    (b)   (5%)

```
for (int x=0; x<20; x++)
    break;
```

2. What is the "exact output" of the following program? Use symbols □ and ↵ to represent blank and empty-line, respectively, if necessary. (10%)

```
#include <iostream>
using namespace std;
main() {
    int n = 4, k = 2;
    cout << ++n << endl;
    cout << n << endl;
    cout << n++ << endl;
    cout << n << endl;
    cout << -n << endl;
    cout << --n << endl;
    cout << n << endl;
    cout << n-- << endl;
    cout << "k=" << n  endl;
    cout << n << k << endl;
    return 0;
}
```

3. Any differences between the following two functions besides their names? (5%)

```
int& f (int &r, int i) {
    r = i*i ;
    return i ;
}
```

```
int& g (int &r, int i) {
    r = i*i ;
    return r ;
}
```

4. Write the output of the following piece of C++ program. (10%)

```
#include <iostream>
void print(int x = 1, int y = 2, int z = 3)
{
  std::cout << x << y << z;
}
int main()
{
  print(), print(4), print(5, 6), print(7, 8, 9);
  return(EXIT_SUCCESS);
}
```

系別：資訊工程學系　　　　科目：程式語言 C++

5. State the differences between the two '=' operators in the following statements.　(10%)
   (a)　`c1 = c2;`
   (b)　`T c1 = c2;`

6. About inheritance, what is the output of the following code?　(10%)

```
#include <iostream>
using std::cout;
class A { public:
  int y;
  virtual void foo1(){cout << "calss A";};
};
class C : virtual public A
{
  public:
  int x;
};
class B : virtual public A {
  public :int x;
  virtual void foo1() {cout<<"I'm B::foo1\n";}
};
class D : public B, public C {};
main(){ C c; B b; D d;
  c.y=1;
  b.y=2; cout<<c.y<<endl; cout<<b.y<<endl;
  d.B::y=3; d.C::y=4;
  cout <<d.y<<endl;
  cout <<d.B::y<<endl;
  cout <<d.C::y<<endl;
}
```

7. The following code is intended to replace all occurrence of substring $sub1$ by a new string nst in the original string str:

```
void replacestr(char *str, char *sub1, char *nst)
{
    int lsub1= strlen(sub1);
    int lstr=strlen(nst);
    int location = findfirst(str, sub1);
    if (location!=-1) {
        movestr(&str[location+lsub1],&str[location+lnstr]);
        replacestr(&str[location+lsub1],sub1, nst);
    }
}
```

here the **strlen** returns the length of the char string, and **findfirst** returns the location of first occurrence of $sub1$ in str. **findfirst**(str, $sub1$) returns -1 if $sub1$ is not found in str. and **movestr** (char *src, char *$dest$) move the substring at src to dst. Will the above implementation generates the desired result? if not, what is the problem and how to fix it?　(10%)

系別：資訊工程學系　　　　　科目：程式語言 C++

8. About STL, write the output of the following C++ program.　　　(10%)

```cpp
#include    <iostream>
#include    <vector>
#include    <iterator>
int main () {
    int array[6]={1,2,3,4,5,6};
    std::vector<int> integers(array, array+6);
    std::ostream_iterator<int> output ( std::cout, "*");
    std::copy( integers.begin()++, integers.end(), output);
    return 0;
}
```

9. About exception handling, what is the output of the following program?　　(10%)

```cpp
void throwException() {
    try {
        cout << " FUN: Function throwException throws an
        exception\n";
        throw exception(); // generate exception
    }
    catch ( exception &caughtException ){
        cout << " FEX: Exception handled in function throwException"
        << "\n Function throwException rethrows exception\n";
        throw; // rethrow exception for further processing
    }
    cout << "after the try/catch\n";
}
int main()
{
    try {
        cout << "\n MAN: main invokes function throwException\n";
        throwException();
        cout << "MAN: This might not print\n";
    }
    catch ( exception &caughtException ){
        cout << "\n\ nMEX: Exception handled in main\n";
    }
    cout << "MAN: Program control continues after catch in main\n";
}
```

10. Fibonacci numbers a_n, $n \geq 0$, are recursively defined as $a_n = a_{n-1} + a_{n-2}$, and $a_0 = 0$, $a_1 = 1$.
    (a) Write a non recursive C++ function with for-loops which receives input parameter n and returns the corresponding Fibonacci number a_n, for $n \geq 0$.　　(5%)
    (b) Rewrite a *recursive* function to compute a_n *without* using *global* variables.　　(10%)