1. Study the following program carefully assuming the necessary files have been #included. The indicate the values of the variables indicated in the spaces provided below, when the end of the main function is reached. If a variable has not been assigned a value write no value and if the variable is not defined in the current scope write undefined.

```c
float f();
float f(int & a, int b=10);
bool g(int, string);

int main()
{
    int x=1,y=2,z=3;
    float a,b,c,d;
    bool flag;

    a = f();
    b = f(x);
    c = f(y,z);
    if (g(y,"comp11")) {
        flag=g(z,"comp11");
    }

    cout << "The end!\n";
    return 0;
}

float f() {
    float d = 5.5;
    return (d);
}

float f(int & a, int b) {
    a+=10;
    return (float(a)/b);
}

bool g(int mult, string s) {
    static int big = 10;

    if ((s == "comp11") && (big <= 10)) {
        big *= mult;
        mult += 1;
        return true;
    } else {
        mult += 10;
        return false;
    }
}
```

Variable Values are:

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>y:</td>
<td></td>
</tr>
<tr>
<td>z:</td>
<td></td>
</tr>
<tr>
<td>m:</td>
<td></td>
</tr>
<tr>
<td>a:</td>
<td></td>
</tr>
<tr>
<td>b:</td>
<td></td>
</tr>
<tr>
<td>s:</td>
<td></td>
</tr>
<tr>
<td>d:</td>
<td></td>
</tr>
</tbody>
</table>
2. For each of the following programs write clearly what the program prints to the screen. You may assume that all appropriate files are included with #include. Make sure to mark all spaces in your output by _

(a) string f(string & a, int b) {
    char c;
    string s;
    s=a;
    if (b>4)
        c = 'R';
    else
        c = 'A' + b;
    a = a + c;
    return s;
}

int main() {
    int x=1;
    string s="CA",t;
    t = f(s,x);
    cout << x << " " << s << " " << t << endl;
}

The Program Prints:

(b) int x=1;

    int f(int a) {
        x=a+3;
        return x;
    }

    int main() {
        int y;
        y=f(x);
        cout << x << " " << y << endl;
    }

The Program Prints:
(c) int x=1;

int f(int & a) {
    a *= 2;
    cout << "f():" << x << " " << a << endl;
    return x;
}

int main() {
    int y=5,z;
    z=f(y);
    cout << "main(): " << x << " " << y << " " << z << endl;
    y=f(x);
    cout << "main(): " << x << " " << y << " " << z << endl;
}

The Program Prints:

(d) int f(int a, int b, int c=10) {
    if ((a>b) && (a>c))
        return a;
    else
        return(b+c);
}

int f(int a, float b=2.0) {
    int c;
    c = int(b)*3;
    return(a+c);
}

int main() {
    int x=3,y=1,R;
    R = f(x,x,x);
    cout << x << " " << y << " " << R << endl;
    R = f(x);
    cout << x << " " << y << " " << R << endl;
    R = f(x,y);
    cout << x << " " << y << " " << R << endl;
}

The Program Prints:
(e) int f(int a) {
    int d,m;
    cout << "entering f(): " << a << endl;
    if (a<10) {
        d=m=a;
    } else {
        d = a%10;
        m = f(a/10);
        if (d>m)
            m = d;
    }
    cout << "exiting f(): " << a << " " << d << " " << m << endl;
    return(m);
}

int main() {
    int x=7913,y;
    y = f(x);
    cout << y << endl;
}

The Program Prints:

(f) class Car {
public:
    float gas;
    Car() { gas=20; }

    void drive(int a) { gas = gas - 0.5 * a; }
    void fill() { gas = 20; }
    float hours() { return(gas/0.5); }
};

int main() {
    Car A,B;
    cout << A.gas << endl;
    A.drive(3);
    A.drive(2);
    cout << A.gas << endl;
    B=A;
    cout << B.hours() << endl;
    B.fill();
    cout << B.hours() << endl;
}

The Program Prints:
3. Write header file beans.h and implementation file beans.cpp and main program to represent the soaking and cooking process of dry beans.

- The beans have a level of softness between 0 (very hard) and 120 (ready to eat).
- Dry beans bought in the shop vary in softness between 0 and 55.
- To prepare the beans one must first soak them overnight (for one or more nights). Soaking can increase softness to a maximum of 110. Each overnight soaking adds 50 to the softness value, unless this passes 110 maximum in which case softness stays at 110.
- Once the beans have softness at least 100 one can cook them. Each minute of cooking adds one to the softness. Cooking can stop when the beans’ softness is 120.

(a) In this part you implement the class itself. The interface for the class beans is as follows:
- A constructor with one integer argument, which defaults to 0, initializing the softness value to the argument value.
- A member function softness() which returns the softness level of the beans.
- A public member function soak() which simulates soaking of the beans. Each call to soak() should simulate the effect of overnight soaking.
- A public boolean member function boil(int M) which simulates boiling of the beans. If the softness is below 100 the function returns false. If the softness is at least 100, the function simulates the effect of cooking for M minutes.
- A public boolean member function ready() which returns true if the softness is at least 120.

Write the code for the header and implementation files of beans.

The header file:

The code file:

(b) In this part you should write a function cook(...) that takes a beans object as argument and simulates the process of cooking. First the function should soak the beans for as many nights as necessary and then it should cook the beans until they are ready. The function should return the cooked beans object once it is ready but should not modify the original object.

The cook function: