Questions

1.证明/证伪

a.

证明：因为$t(n) \in O(g(n))$，于是存在正常数c和n_0，使得对于所有的$n \geq n_0$，有$t(n) \leq cg(n)$。

于是，对于任意的$n \geq n_0$，有$g(n) \geq \frac{1}{c}t(n)$。

即，存在正常数$c' = \frac{1}{c}$和$n_0' = n_0$，使得对于所有的$n \geq n_0'$，有$g(n) \geq c't(n)$。即，$g(n) \in \Omega(t(n))$。

b.

反例：设$t(n) \in \Theta(n)$，$g(n) \in \Theta(n^2)$，则$t(n) \in O(g(n))$但$t(n) \notin \Theta(\alpha g(n))$。

因此，$\Theta(\alpha g(n)) \neq O(g(n))$。

c.

证明：设$t(n) \in \Theta(g(n))$，则存在正常数c_1和c_2，使得对于所有的$n \geq n_0$，有$c_1 g(n) \leq t(n) \leq c_2 g(n)$。

因为存在正常数c_1和n_0，使得对于所有的$n \geq n_0$，有$t(n) \geq c_1 g(n)$，所以有$t(n) \in \Omega(g(n))$。

同理，有$t(n) \in O(g(n))$。

综上，$t(n) \in O(g(n)) \cap \Omega(g(n))$。

同理可证，当$t(n) \in O(g(n)) \cap \Omega(g(n))$时，有$t(n) \in \Theta(g(n))$。

因此，$\Theta(g(n)) = O(g(n)) \cap \Omega(g(n))$。

d.

证明：由c题证明可知，$t(n) \in O(g(n)) \cap \Omega(g(n))$与$t(n) \in \Theta(g(n))$等价。

$t(n)$和$g(n)$只可能有三种关系，即$t(n) \in O(g(n))$，$t(n) \in \Omega(g(n))$，$t(n) \in \Theta(g(n))$，所以命题成立。

2.计算时间复杂度

a.

$O(\sqrt{n})$

b.

$O(n^3)$

3.计算时间复杂度

a.

最好情况：$O(1)$，最坏情况：$O(n)$，平均情况：$O(\log(n))$

b.

最好情况：$O(n)$，最坏情况：$O(n^2)$，平均情况：$O(\log(n))$

4.解递归式

a.

$T(n) = n + T(n-1) = n + (n-1) + T(n-2) = ... = n + (n-1) + (n-2) + ... + 1 + 1 = \frac{n \times (n+1)}{2} + 1$

b.

$T(n) = 4T(n-1) = 4^2T(n-2) = ... = 4^{n-1}T(1) = 5 \times 4^{n-1}$

c.

$T(n) = n + T(\frac{n}{3}) = n + \frac{n}{3} + T(\frac{n}{3^2}) = ... = n + \frac{n}{3} + \frac{n}{3^2} + ... + 3 + 1 = \frac{3^{k+1}-1}{2}$

解题思路

Q1

从经典递归问题爬楼梯问题出发进行改进，爬楼梯问题的递归公式为：

```
f(n) = f(n-1) + f(n-2)
```

本题需要的参数除了要爬的步数，还有剩余卡路里数，因此递归公式为：

```
f(m, n) = f(m-1, n-1) + f(m-2, n-3)
```

再考虑递归边界条件。

- 如果要爬的步数或卡路里数不为正，无法完成，返回0
- 如果要爬的步数为1且卡路里足够，只有1种方法
- 如果要爬的步数为2且卡路里足够，只有2种方法

代码实现分别如下：

```
if (m <= 0 || n <= 0)
        return 0;


if (m == 1 && n >= 1)
    return 1;


if (m == 2 && n >= 3)
    return 2;
```

综合以上思路即可得到本题核心函数代码：

```
int Q1(int m, int n)
{
    if (m <= 0 || n <= 0)
        return 0;

    if (m == 1 && n >= 1)
        return 1;

    if (m == 2 && n >= 3)
        return 2;

    return Q1(m - 1, n - 1) + Q1(m - 2, n - 3);
}
```

Q2

在Q1基础上改进。

改进函数返回值

Q1函数的返回值为卡路里充足情况下爬到m阶楼梯的方法数，本题则需要返回：

- 卡路里充足情况下爬到m阶楼梯，且尽可能多消耗卡路里的方法数
- 最大消耗卡路里数

因此改为返回std::pair<int, int>类型，第一个元素为方法数，第二个元素为最大消耗卡路里数。

改进递归公式

递归公式需要考虑“尽可能消耗卡路里“这一条件。Q1的递归公式是m-1步和m-2步的方法数之和，Q2需要比较这两种情况爬到m阶的消耗卡路里数，选择消耗卡路里数更多的那种方法。

代码实现如下：

```
int k1 = Q2(m - 1, n - 1).second + 1;
int k2 = Q2(m - 2, n - 3).second + 3;

if (k1 > k2)
{
    return Q2(m - 1, n - 1);
}
else if (k1 < k2)
{
    return Q2(m - 2, n - 3);
}
else
{
    return pair<int, int>(Q2(m - 1, n - 1).first + Q2(m - 2, n - 3).first, k1);
}
```

改进递归边界条件

边界条件与Q1不同之处仅在于返回值，考虑“尽可能消耗卡路里”的条件和函数返回值类型，代码实现如下：

```
if (m <= 0 || n <= 0)
    return pair<int, int>(0, 0);

if (m == 1 && n >= 1)
    return pair<int, int>(1, 1);

if (m == 2 && n >= 3)
    return pair<int, int>(1, 3);
```

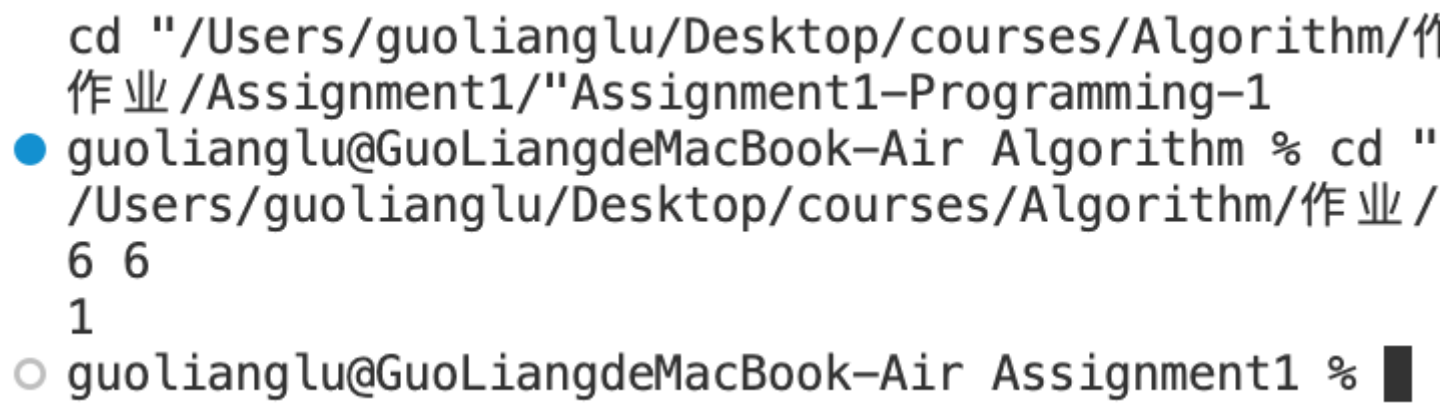
综合以上思路即可得到本题核心函数代码：

```
pair<int, int> Q2(int m, int n)
{
    if (m <= 0 || n <= 0)
        return pair<int, int>(0, 0);

    if (m == 1 && n >= 1)
        return pair<int, int>(1, 1);

    if (m == 2 && n >= 3)
        return pair<int, int>(1, 3);

    int k1 = Q2(m - 1, n - 1).second + 1;
    int k2 = Q2(m - 2, n - 3).second + 3;

    if (k1 > k2)
    {
        return Q2(m - 1, n - 1);
    }
    else if (k1 < k2)
    {
        return Q2(m - 2, n - 3);
    }
    else
    {
        return pair<int, int>(Q2(m - 1, n - 1).first + Q2(m - 2, n - 3).first, k1);
    }
}
```

输出结果时，只需取出函数返回的pair中的第一个元素即可。

运行结果

Q1 test case 1:



Q1 test case 2:

```
guolianglu@GuoLiangdeMacBook-Air Assignment1 % cd '
 "/Users/guolianglu/Desktop/courses/Algorithm/作业/
3 6
3
guolianglu@GuoLiangdeMacBook-Air Assignment1 %
```

Q1 test case 3:

```
guolianglu@GuoLiangdeMacBook-Air Assignment1 % cd "/Use
 "/Users/guolianglu/Desktop/courses/Algorithm/作业/Assi
-5 7
0
guolianglu@GuoLiangdeMacBook-Air Assignment1 %
```

Q2 test case 1:

```
guolianglu@GuoLiangdeMacBook-Air Assignment1 % cd "/Users/guoli
 "/Users/guolianglu/Desktop/courses/Algorithm/作业/Assignment1/"
7 6
0
guolianglu@GuoLiangdeMacBook-Air Assignment1 %
```

Q1 test case 2:

```
guolianglu@GuoLiangdeMacBook-Air Assignment1 % cd "/Users/g
 "/Users/guolianglu/Desktop/courses/Algorithm/作业/Assignme
3 6
2
guolianglu@GuoLiangdeMacBook-Air Assignment1 %
```