Problem

Duda the rock monster lives in the enchanted forest and has collected N energy stones for lunch. Since he has a smal mouth, he eats energy stones one at a time. Some stones are tougher than others! The i-th stone takes him Si second to eat.

Duda eats energy stones to get energy. Different stones give him different amounts of energy. Furthermore, the ston s lose energy over time. The i-th stone initially contains Ei units of energy and will lose Li units of energy each seco d. When Duda starts to eat a stone, he will receive all the energy the stone contains immediately (no matter how muc time it takes to actually finish eating the stone). The stone's energy stops decreasing once it hits zero.

What is the largest amount of energy Duda could receive from eating his stones?

Input

The first line of the input gives the number of test cases, T. T test cases follow. Each test case starts with a line conta ning the integer N, the number of energy stones Duda has. Then, there are N more lines, the i-th of which contains th three integers Si, Ei and Li, as described above.

Output

For each test case, output one line containing Case #x: y, where x is the test case number (starting from 1) and y is th maximum amount of energy Duda could receive from eating stones.

Limits

Time limit: 30 seconds per test set.

Memory limit: 1GB.

$1 \leq T \leq 100$.

$1 \leq N \leq 100$.

$1 \leq Si \leq 100$.

$1 \leq Ei \leq 105$.

$0 \leq Li \leq 105$.

Test set 1 (Visible)

All stones take the same amount of time to eat. That is: Si = Sj for all i and j.

Test set 2 (Hidden)

There are no additional constraints beyond the general Limits.

Sample

Input

Output

```
3
4
20 10 1
5 30 5
100 30 1
5 80 60
3
10 4 1000
10 3 1000
10 8 1000
2
```

12 300 50
5 200 0

Case #1: 105
Case #2: 8
Case #3: 500

In Sample Case #1, there are N = 4 stones. One possible order Duda can eat stones is:
Eat the fourth stone. This takes 5 seconds and gives him 80 units of energy.
Eat the second stone. This takes 5 more seconds and gives him 5 units of energy (the second stone started with 30 en rgy, and over 5 seconds, has lost 25 units of energy).
Eat the third stone. This takes 100 more seconds and gives him 20 units of energy (the third stone started with 30 en rgy, and over 10 seconds, has lost 10 units of energy).
Eat the first stone. This takes 20 more seconds and gives him 0 units of energy (the first stone started with 10 units o energy, and over 110 seconds, has lost all of its energy).
This gives him 105 units of energy, which is the best he can do. So the answer is 105.

In Sample Case #2, there are N = 3 stones. No matter which stone Duda eats, the other two will have no energy left nce he is done eating. So he should eat the third stone, giving him 8 units of energy.

In Sample Case #3, there are N = 2 stones. Duda can:
Eat the first stone. This takes 12 seconds and gives him 300 units of energy.
Eat the second stone. This takes 5 seconds and gives him 200 units of energy (the second stone does not lose any en rgy over time!).
So the answer is 500.

Note that Sample Cases #1 and #3 will not appear in the Visible dataset. Consequently, sample input is not included s a test case for your submissions, and hence any attempts that fail to produce correct output will accrue a penalty. e recommend you to run a manual test in the editor before submitting the attempt.

// In the name of God

```
#include <iostream>
#include <algorithm>
#include <fstream>
#include <vector>
#include <deque>
#include <assert.h>
#include <queue>
#include <stack>
#include <set>
#include <map>
#include <stdio.h>
#include <string.h>
#include <utility>
#include <math.h>
#include <bitset>
#include <iomanip>
#include <complex>
```

```
using namespace std;

#define rep(i, a, b) for (int i = (a), i##_end_ = (b); i < i##_end_; ++i)
#define debug(...) fprintf(stderr, __VA_ARGS__)
#define mp make_pair
#define x first
#define y second
#define pb push_back
#define SZ(x) (int((x).size()))
#define ALL(x) (x).begin(), (x).end()

template<typename T> inline bool chkmin(T &a, const T &b) { return a > b ? a = b, 1 : 0; }
template<typename T> inline bool chkmax(T &a, const T &b) { return a < b ? a = b, 1 : 0; }
template<typename T> inline bool smin(T &a, const T &b)   { return a > b ? a = b : a;    }
template<typename T> inline bool smax(T &a, const T &b)   { return a < b ? a = b : a;    }

typedef long long LL;

const int N = (int) 2e4 + 4, mod = (int) 0;
int s[N], e[N], l[N], o[N];
int cmp(int x, int y) {
 int xe = e[x] + e[y] - l[y] * s[x];
 int xy = e[x] + e[y] - l[x] * s[y];
 return xe > xy;
}
int dp[N], odp[N];
int main() {
 int tc;
 cin >> tc;
 for (int tt = 1; tt <= tc; ++tt) {
  cout << "Case #" << tt << ": ";
  int n;
  cin >> n;
  for (int j = 0; j < n; ++j) {
   cin >> s[j] >> e[j] >> l[j];
   o[j] = j;
  }
  memset(dp, 0, sizeof dp);
  memset(odp, 0, sizeof odp);
  sort(o, o + n, cmp);
  int sum = 0;
  for (int i = 0; i < n; ++i) {
   int j = o[i];
   sum += s[j];
   for (int secs = 0; secs <= sum; ++secs) odp[secs] = dp[secs];
   for (int secs = s[j]; secs <= sum; ++secs) {
    int en = e[j] - (secs - s[j]) * l[j];
    if (en <= 0) continue;
    dp[secs] = max(dp[secs], odp[secs - s[j]] + en);
   }
  }
  int res = 0;
```

```
  for (int j = 0; j < N; ++j) res = max(res, dp[j]);
  cout << res << '\n';
 }
}
```