```cpp
#include<bits/stdc++.h>
using namespace std;
typedef long long ll;
const int MAX_N = (1 << 21) - 1;
const int inf = numeric_limits<int>::max();

ll all[MAX_N];
ll part[MAX_N];

// call init(0,0,n) to init
// node k => [l,r)
void init(int k,int l,int r){
  if(r - l == 1){
    all[k] = 0; // init leaf
    part[k] = 0;
  }else{
    int chl = k*2+1;
    int chr = k*2+2;
    all[k] = 0; // init internal-node
    part[k] = 0;
    init(chl, l, (l+r)/2);
    init(chr, (l+r)/2, r);
  }
}

// call add(a,b,x,0,0,n) to add
// node k => [l,r)
void add(int a,int b,int x,int k,int l, int r){
  // not cross
  if(r <= a || b <= l)return;
  else if(a <= l && r <= b){
    // [a,b) contain [l,r)
    all[k] += x;
  } else{
    // otherwise
    part[k] += x * (min(b,r) - max(a,l));
    int chl = k*2+1;
    int chr = k*2+2;
    add(a,b,x,chl,l,(l+r)/2);
    add(a,b,x,chr,(l+r)/2,r);
  }
}

// call sum(a,b,0,0,n) to calc sum [a,b)
// node k => [l,r)
ll sum(int a,int b,int k,int l,int r){
  if(r <= a || b <= l) return 0;
  else if(a <= l && r <= b){
    // [a,b) contain [l,r)
    return all[k] * (r - l) + part[k];
  } else{
    // otherwise
    ll res = all[k] * (min(b,r) - max(a,l));
    int chl = k*2+1;
    int chr = k*2+2;
    res += sum(a,b,chl,l,(l+r)/2);
    res += sum(a,b,chr,(l+r)/2,r);
    return res;
  }
}

int main(void){
  int n,q;
  cin >> n >> q;
  init(0,0,n);
  for(int i = 0;i < q;++i){
    int c;
    cin >> c;
    if(c == 0){
      int s,t,x;
```

```
      cin >> s >> t >> x;
      add(s-1,t,x,0,0,n);
    }else{
      int s,t;
      cin >> s >> t;
      cout << sum(s-1,t,0,0,n) << endl;
    }
  }
  return 0;
}
```