void init()

{

memset(aMax, -inf, sizeof(aMax)), memset(aMin, inf, sizeof(aMin));

memset(bMax, -inf, sizeof(bMax)), memset(bMin, inf, sizeof(bMin));

for(int i = n - 1; i >= 0; --i)

{

aMax[i] = aMin[i] = a[i];

bMax[i] = bMin[i] = b[i];

for(int j = 0; j <= N; ++j)//换句话说，就是这些二进制状态，只需要枚举二进制的每一位从1到0，就能够得到所有的前缀的最大值

{

if(!(i>>j&1) && (i|(1<<j)) < n)

{

aMax[i] = max(aMax[i], aMax[i|(1<<j)]);

aMin[i] = min(aMin[i], aMin[i|(1<<j)]);

bMax[i] = max(bMax[i], bMax[i|(1<<j)]);

bMin[i] = min(bMin[i], bMin[i|(1<<j)]);

}

}

}

}

ll c[M + 5];

void work()

{

scanf("%d", &n);

for(int i = 0; i < n; ++i) scanf("%d", &a[i]);

for(int i = 0; i < n; ++i) scanf("%d", &b[i]);

init();

c[n - 1] = 1ll \* a[n - 1] \* b[n - 1];

for(int i = n - 2; i >= 0; --i)

{

c[i] = c[i + 1];

c[i] = max(c[i], 1ll \* aMin[i] \* bMin[i]);

c[i] = max(c[i], 1ll \* aMax[i] \* bMax[i]);

c[i] = max(c[i], 1ll \* aMax[i] \* bMin[i]);

c[i] = max(c[i], 1ll \* aMin[i] \* bMax[i]);

// printf("c[%d] = %lld\n", i, c[i]);

}

ll s = 0;

for(int i = 0; i < n; ++i) s = (s + c[i] % mod) % mod;

s = (s % mod + mod) % mod;

printf("%lld\n", s);

}

int main()

{

int T; scanf("%d", &T);

for(int ca = 1; ca <= T; ++ca)

{

work();

}

return 0;

}