Source Code -

```
     const int high = 100007 ;
     int dp[high][ 2 ], upper[high], n;

     struct Node
     {
          int p, q;
          bool operator < (const Node &temp)
          const { return p == temp.p ? q < temp.q : p <
temp.p; }
     }

     node[high];

     int calculate (int now, int prev)
     {
          return op :: max(node[now].q - node[prev].q, 0 ) -
std ::max(node[now].p - node[prev].q, 0 );
     }

     int main()
     {
          scanf ( "%d" , &n);
          for ( int i = 1 ; i <= n; i++)
               scanf ( "%d%d" , &node[i].p, &node[i].q);
          std ::sort(node + 1 , node + 1 + n);
          for ( int i = 1 ; i <= n; i++)
          {
               if (i < n)
               {
                    dp[i][ 0 ] = dp[i - 1 ][ 0 ] + calc(i,
upper[i - 1 ]);
                    upper[i] = node[upper[i - 1 ]].s >
node[i].s ? upper[i - 1 ] : i;
               }
               if (i > 1 )
               dp[i][ 1 ] = std ::max(dp[i - 1 ][ 1 ] +
calc(i, upper[i - 1 ]), dp[i - 2 ][ 0 ] + calc(i, upper[i -
2 ]));
          }
```

```
        printf ( "%d\n" , std ::max(dp[n - 1 ][ 0 ], dp[n]
[ 1 ]));
        return 0 ;
    }
```