Day 2 – Task 2

Boats

The task can be solved using the following greedy algorithm. During the process we will maintain the set of tied boats S. At each step we try to insert one boat into the set, going from left to right, relative to the ring positions. We initialize S with boat tied to the leftmost ring, so that it is positioned maximally to the left. Let l and r be the coordinates of the endpoints of the last inserted boat and l_i and p_i the length and the ring position of the boat we are currently processing. If $p_i = r$, then we insert i^{th} boat to the set, positioning it to the leftmost possible position. If $p_i < r$ then we check if $l_i < r-l$ and if it holds then we delete last inserted boat from the set and insert the ith boat, at the leftmost possible position. If it does not hold then we just skip the ith boat. This solution is optimal, which can easily be proved by induction.

There are also other solutions based on dynamic programming approach.

Example solution in Pascal:

Day 2 – Task 2

```
const
  maxN = 10000;

type
  TBoat = record
    p, l : Longint;
  end;

var
  n, m : Integer;
  c : array[1..maxN] of TBoat;


procedure Sort;
var
  tmp : TBoat;

  procedure QuickSort(l, r : Integer);
  var
    i, j : Integer;
    p : Longint;
  begin
    i := l;
    j := r;
    p := c[(l + r) div 2].p;
    repeat
      while c[i].p < p do inc(i);
      while p < c[j].p do dec(j);
      if i <= j then
      begin
        tmp := c[i];
        c[i] := c[j];
        c[j] := tmp;
        inc(i);
        dec(j);
      end;
    until i > j;
    if l < j then QuickSort(l, j);
    if i < r then QuickSort(i, r);
  end;

begin
  QuickSort(1, n);
end;


function Max(a, b : Longint) : Longint;
begin
  if a > b then
    Max := a
  else
    Max := b
end;


procedure Solve;
var
  i : Integer;
  l, r, t : Longint;
begin
  Sort;

  m := 1;
  r := c[1].p;

  for i := 2 to n do
    if c[i].p >= r then
    begin
```

Day 2 – Task 2

```
      l := r;
      r := Max(c[i].p, r + c[i].l);
      inc(m);
    end else
    begin
      t := Max(c[i].p, l + c[i].l);
      if t < r then
        r := t;
    end;
end;

begin
  Readln(n);
  for i := 1 to n do
    Readln(c[i].l, c[i].p);
  Solve;
  Writeln(m);
end.
```