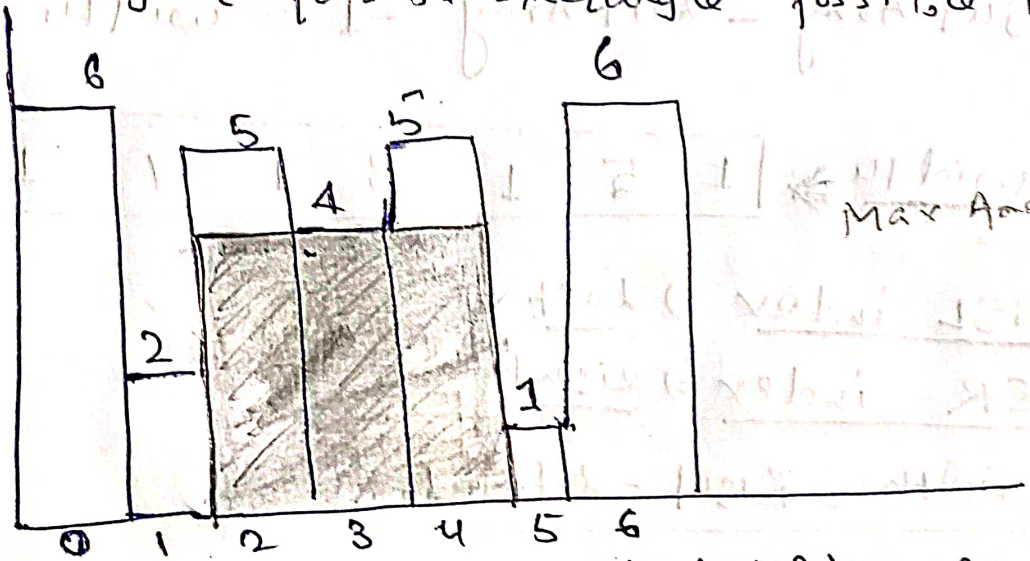
Maximum Area Histogram (MAH)

Problem Statement:-

find the largest rectangular area possible in a given histogram where the largest rectangle can be made of a number of contiguous bars. for simplicity, assume that all bar have same width and width is 1 unit.

for example, consider the following histogram with 7 bars of heights {6, 2, 5, 4, 5, 1, 6}

The largest possible rectangle possible is 12



Max Area = 3 × 4 = 12

- Check whose or which building we can expand
- A building can be expanded into ~~a~~ other building only when the height of other building is greater than or equal to the current building.

Arr :

0	1	2	3	4	5	6
6	2	5	4	5	1	6

NSR → Right index

1	5	5	5	5	7	7

NSL → Left index

-1	-1	1	1	3	-1	5

- Right Array - Left Array -1 = width of rectangle

↳ width →

1	5	1	3	1	7	1

① NSL index ⇒ Left

② NSR index ⇒ Right

③ width = Right - Left - 1.

```
Area[i] = arr[i] * width[i]
```

↳ Area 2 6 10 5 12 5 7 6

↓

Return 12 Ans

Code to find left (NSL) Index:

```
Vector <int> left;
Stack < pair <int, int>> S;
int    pseudo_index = -1; // assume it on left most
for (int i=0; i<n; i++)
{
   if (S.size()==0)
   {
     left.push_back (pseudo_index);
   }
  else if (S.size()>0 && S.top().first < arr[i])
   {left.push_back(S.top().second);
   }
  else if (S.size()>0 && S.top().first >= arr[i])
   {
    while(S.top()>0 && S.top().first>=arr[i])
      { S.pop();
      }
    if(S.size()==0)
     left.push_back(pseudo_index);
    else
      left.push_back(S.top().second);
    }
    S.push(arr[i]);
}  return left;
```

TO find right (NSR Index)

Code:

- Loop chise
- vector Name right
- Pseudo-index=-1

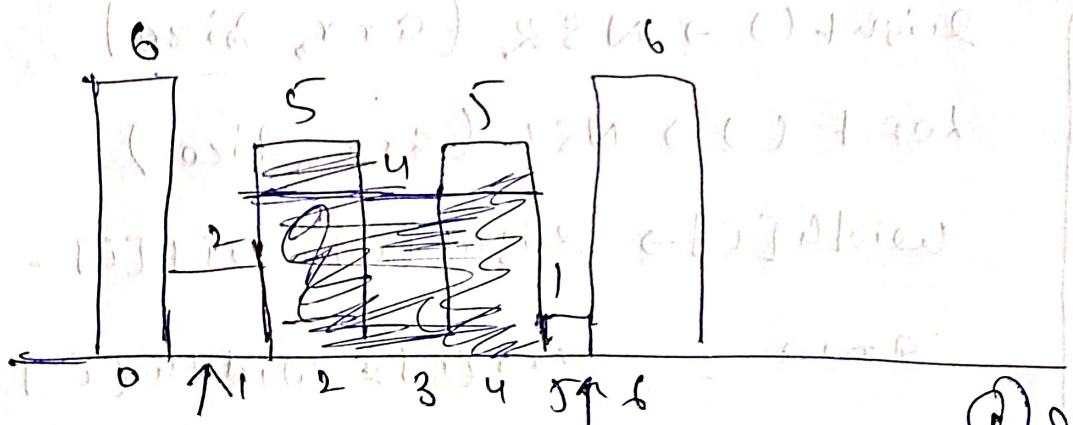
```
vector<int> right;
stack<pair<int,int>> s1;
int pseudo_index=-1;
for(int i=size-1; i>=0; i--)
{
  if(s1.size()==0)
  {
    right.push_back(pseudo_index);
  }
  else if(s1.size()>0 && s1.top().first<arr[i])
  {
    right.push_back(s1.top().second);
  }
  else if(s1.size()>0 s.top().first>=arr[i])
  {
  while(s.top()>0 && s1.top().first>=arr[i])
    {
      s1.pop();
    }
  if(s1.size==0)
  right.push_back(pseudo_index);
  else
  right.push_back(s.top().second);
  }
  s.push(arr[i])
reverse(right.begin(), right.end())
  return right;
}
```

```
for (i=0; i<size; i++)
width [i] = right [i] - left[i] - 1;
}
for (i=0; i<size; i++)
{
area[i] = arr[i] * width[i]
|| area[i] → max
Max (area.begin(), area.end())
}
return max;
```

3

Sum up:

arr : 6 2 5 4 5 1 6 → height of bars



Left = NSL index

NSR = Right index

① width
② Left
③ width vector
④ area vector
Vs
Max of Area vector