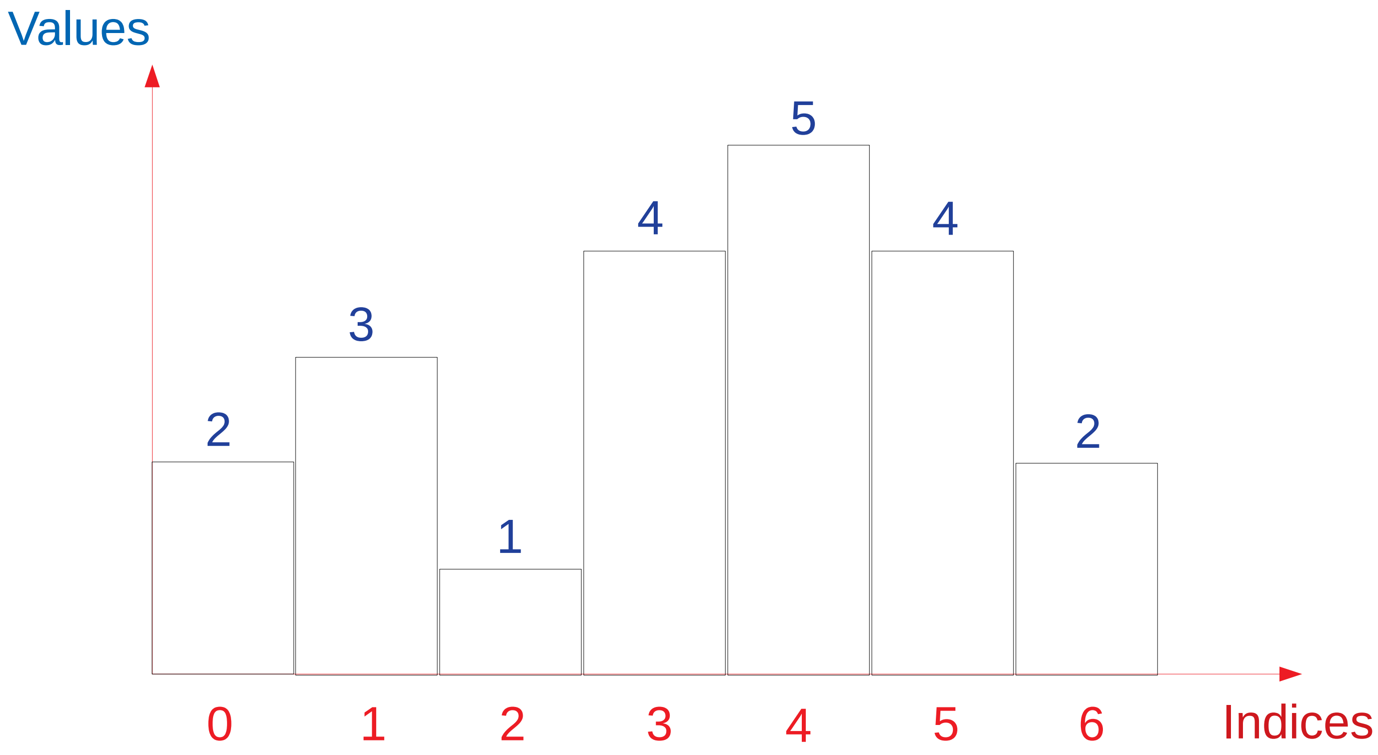
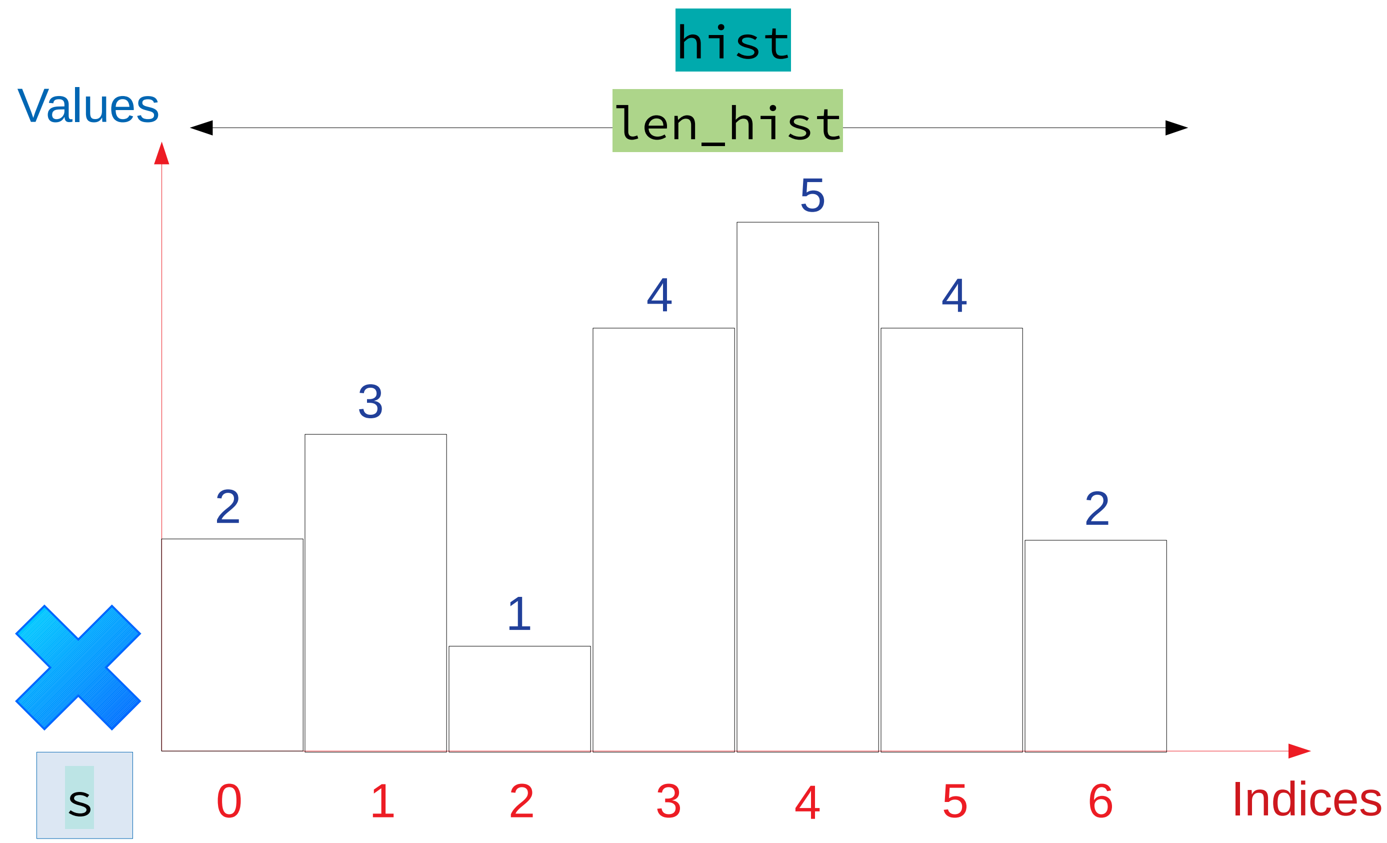
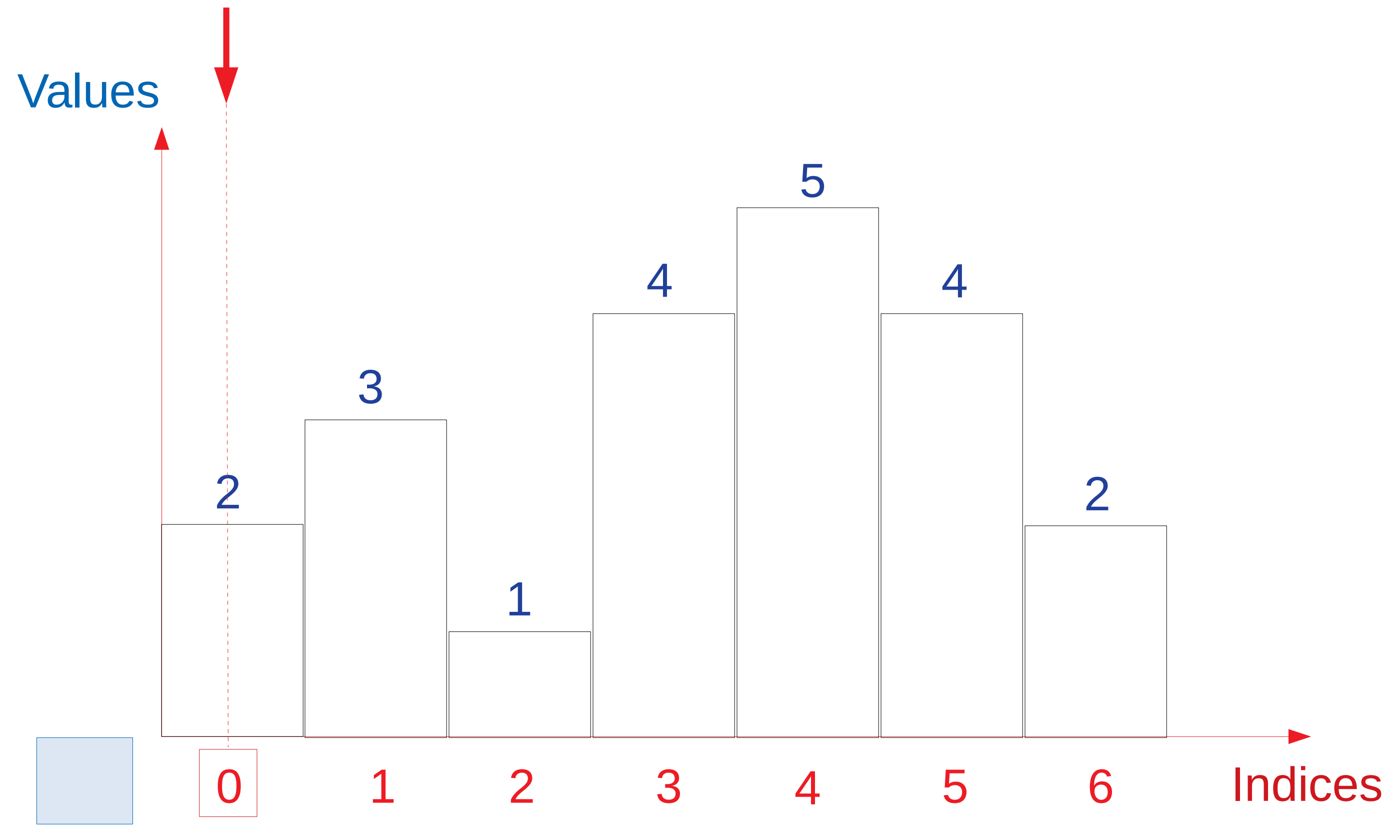
hist



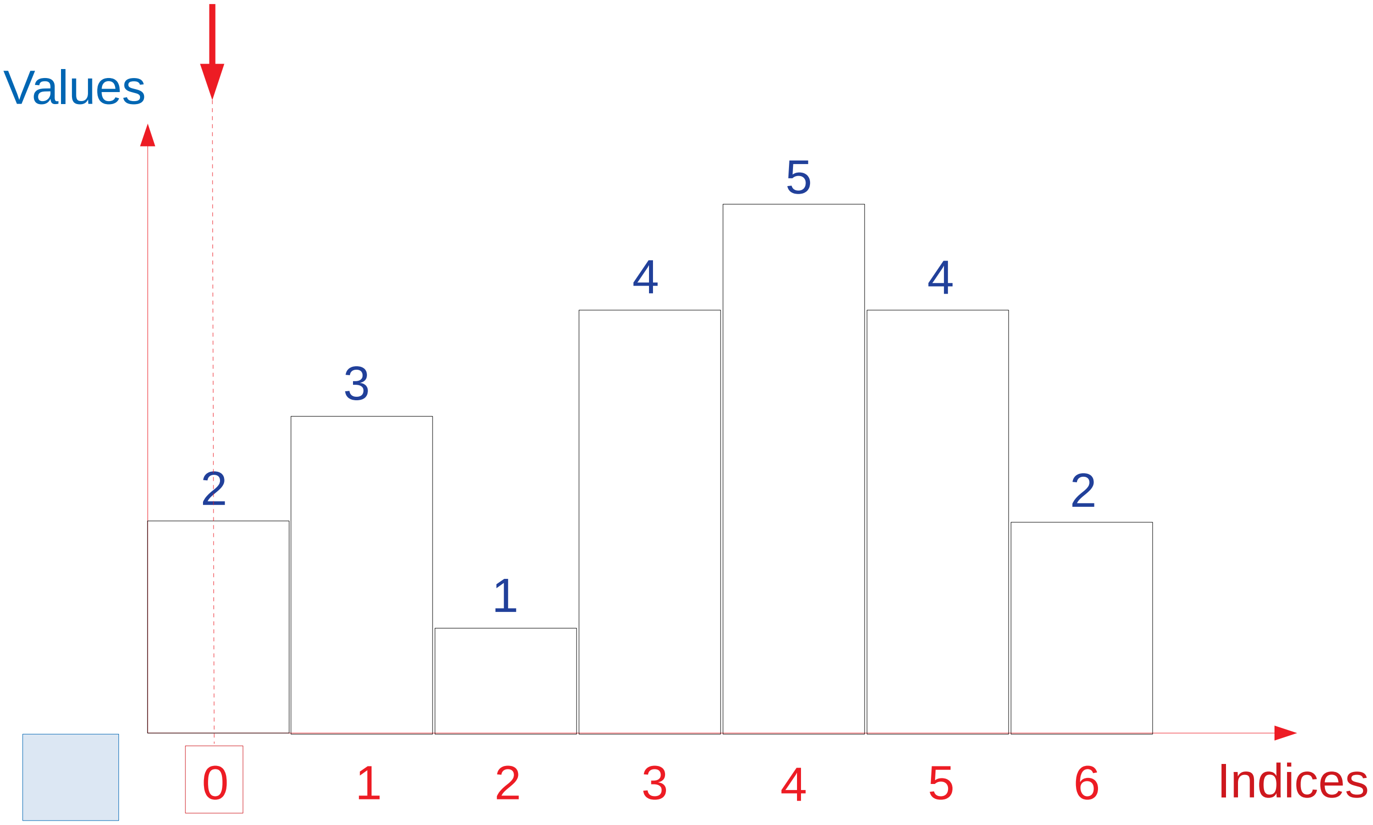
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



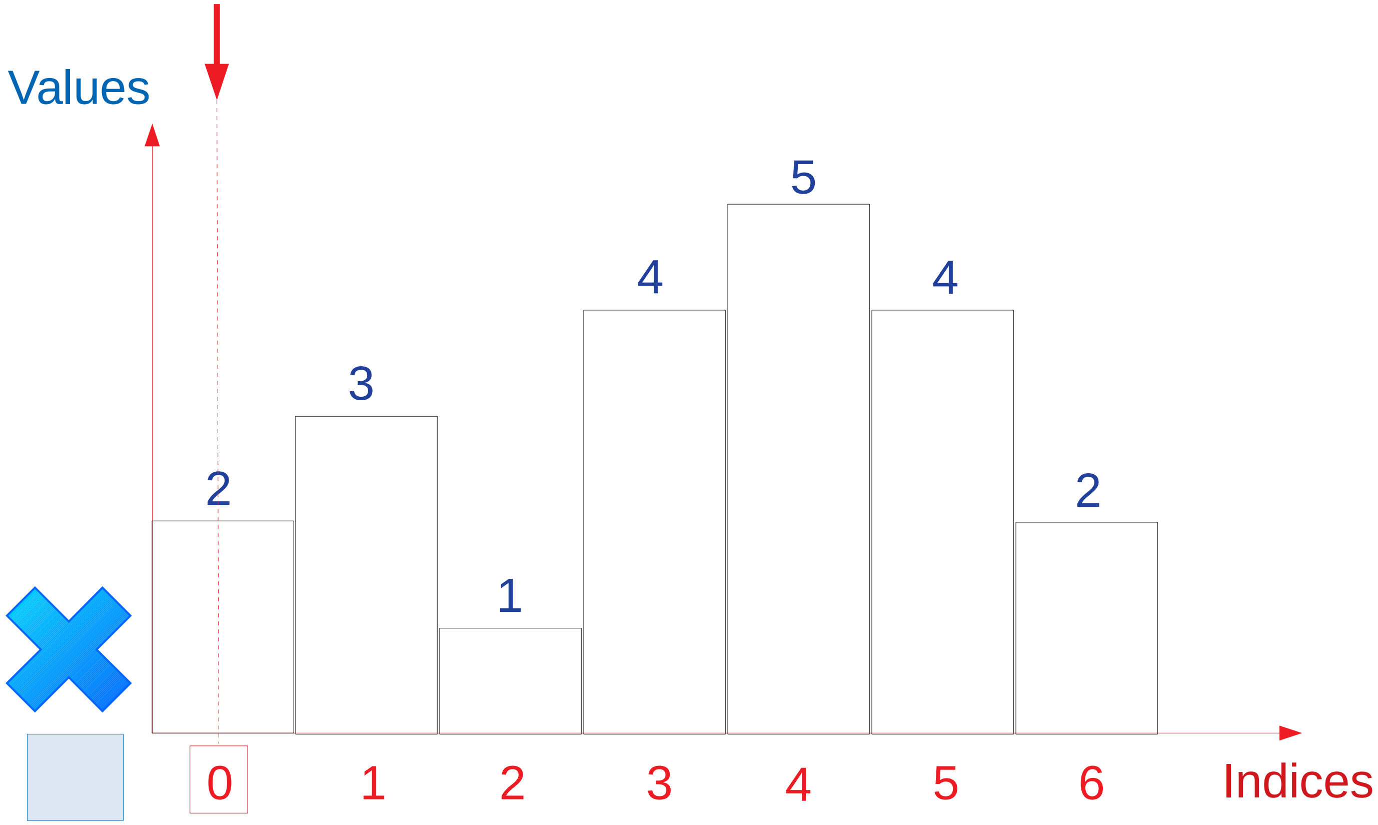
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



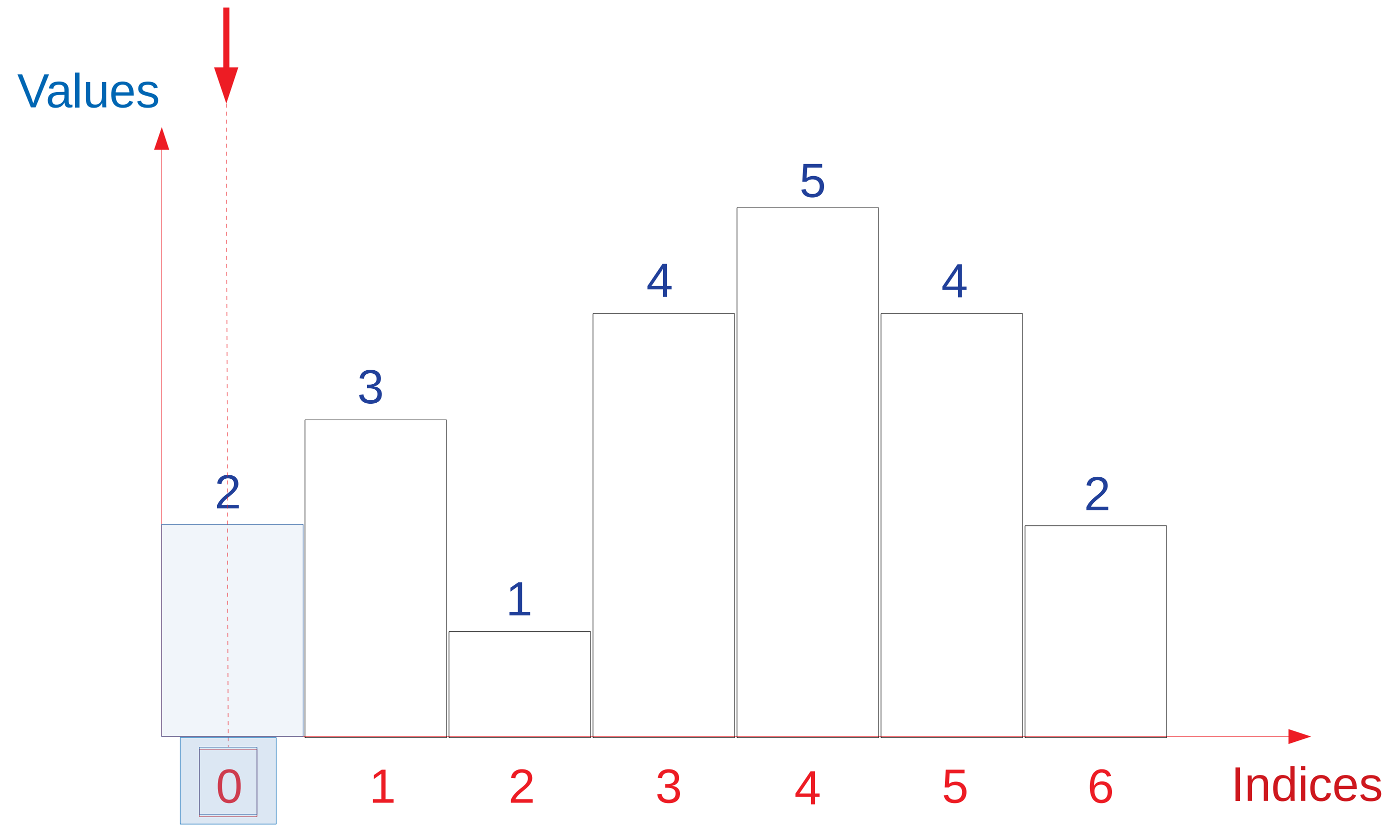
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



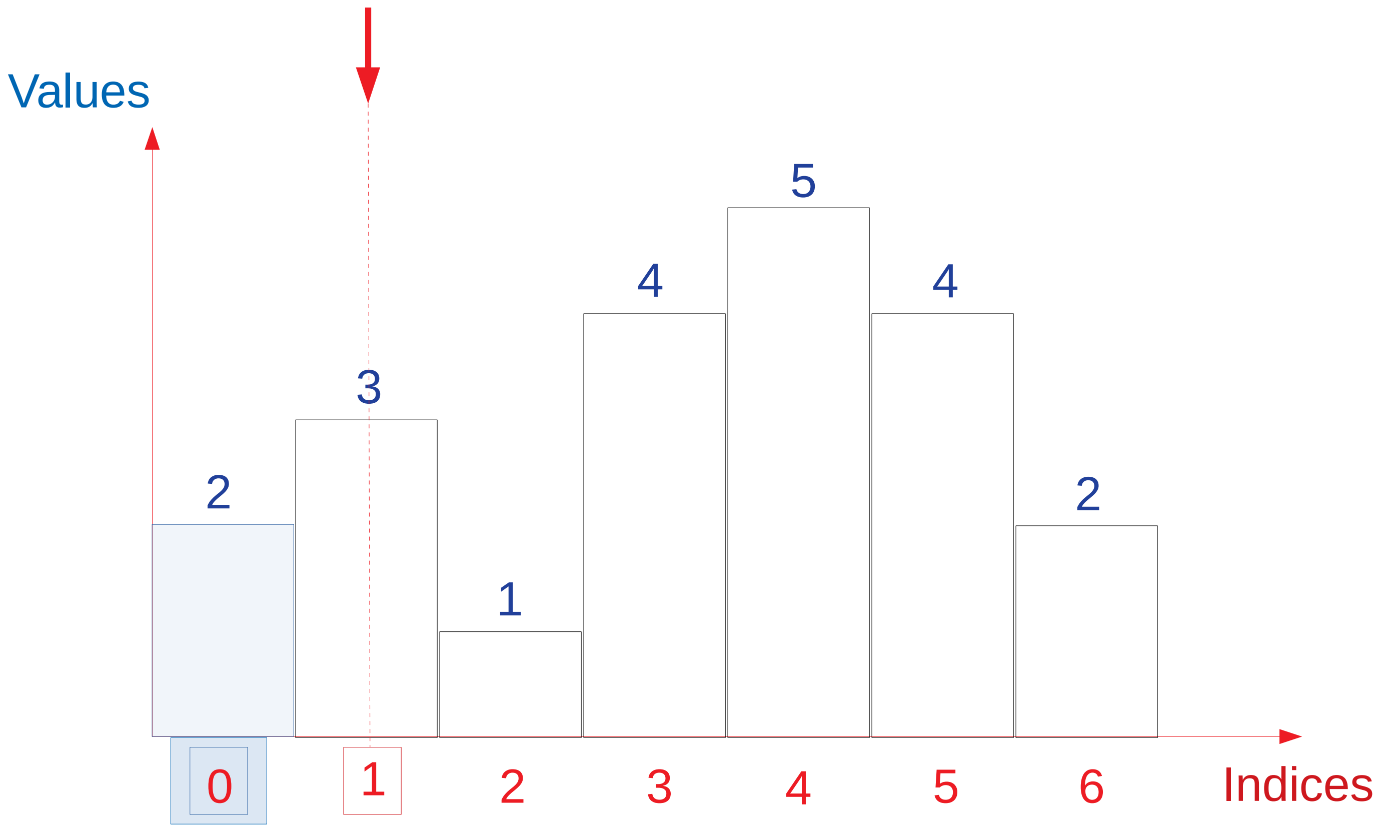
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



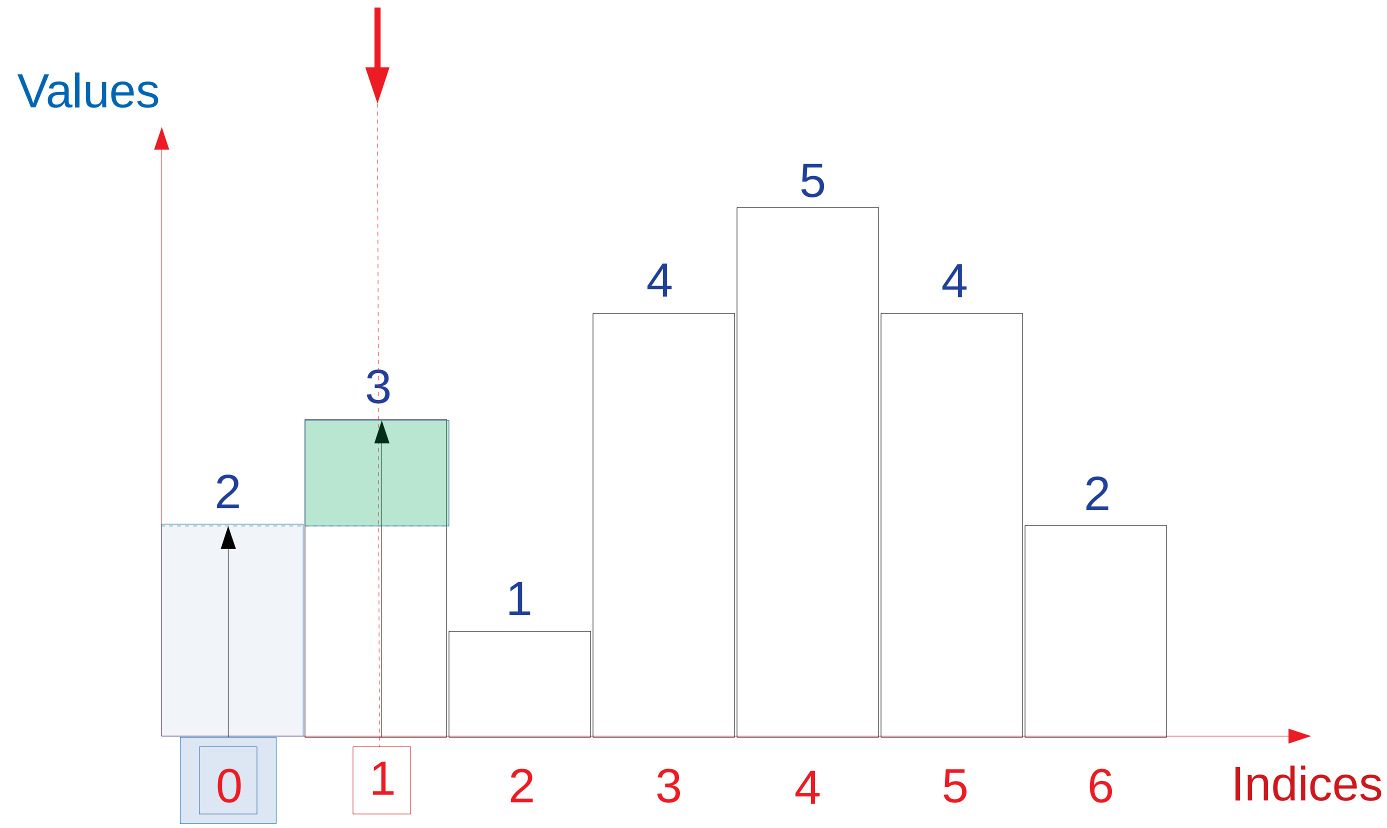
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



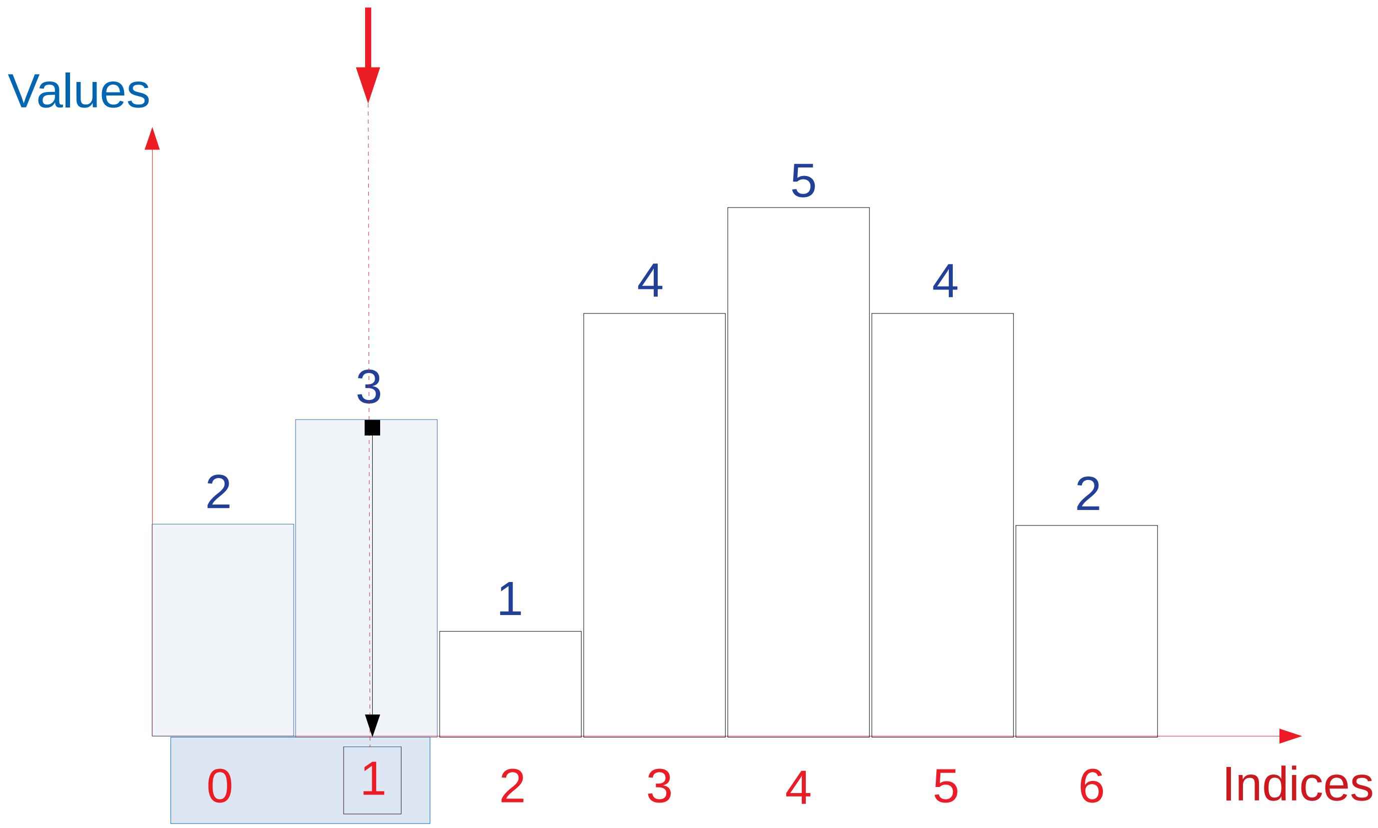
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



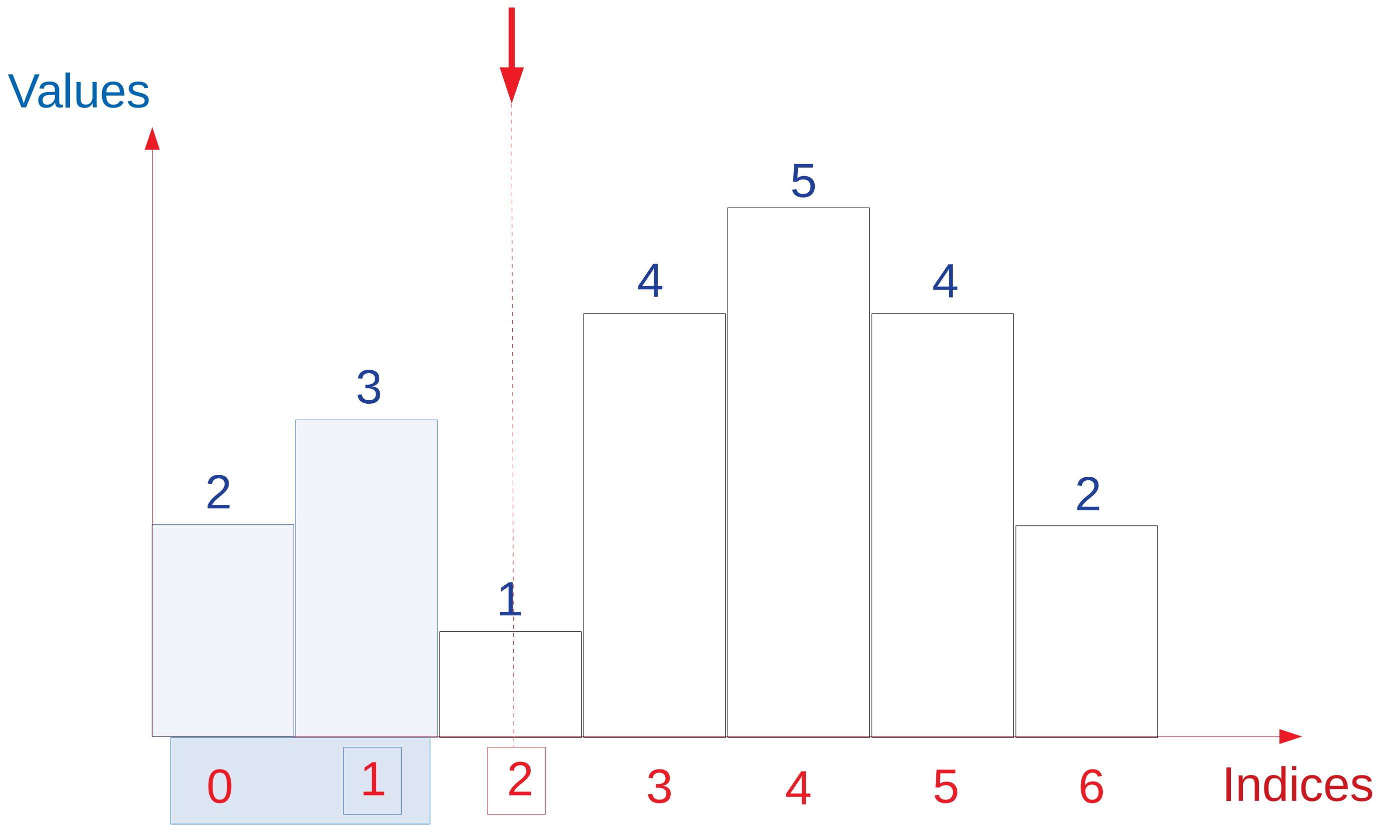
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
      (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                  else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



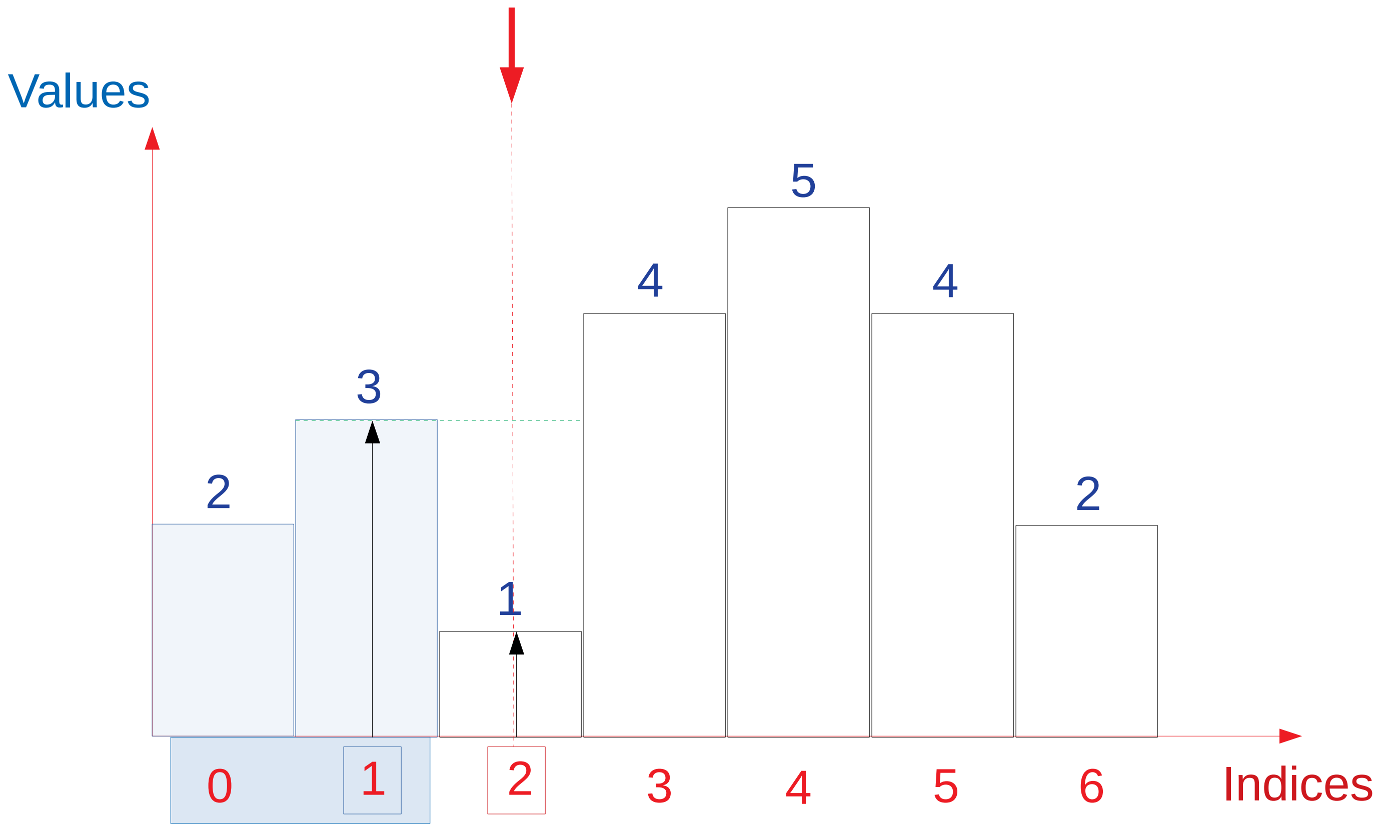
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



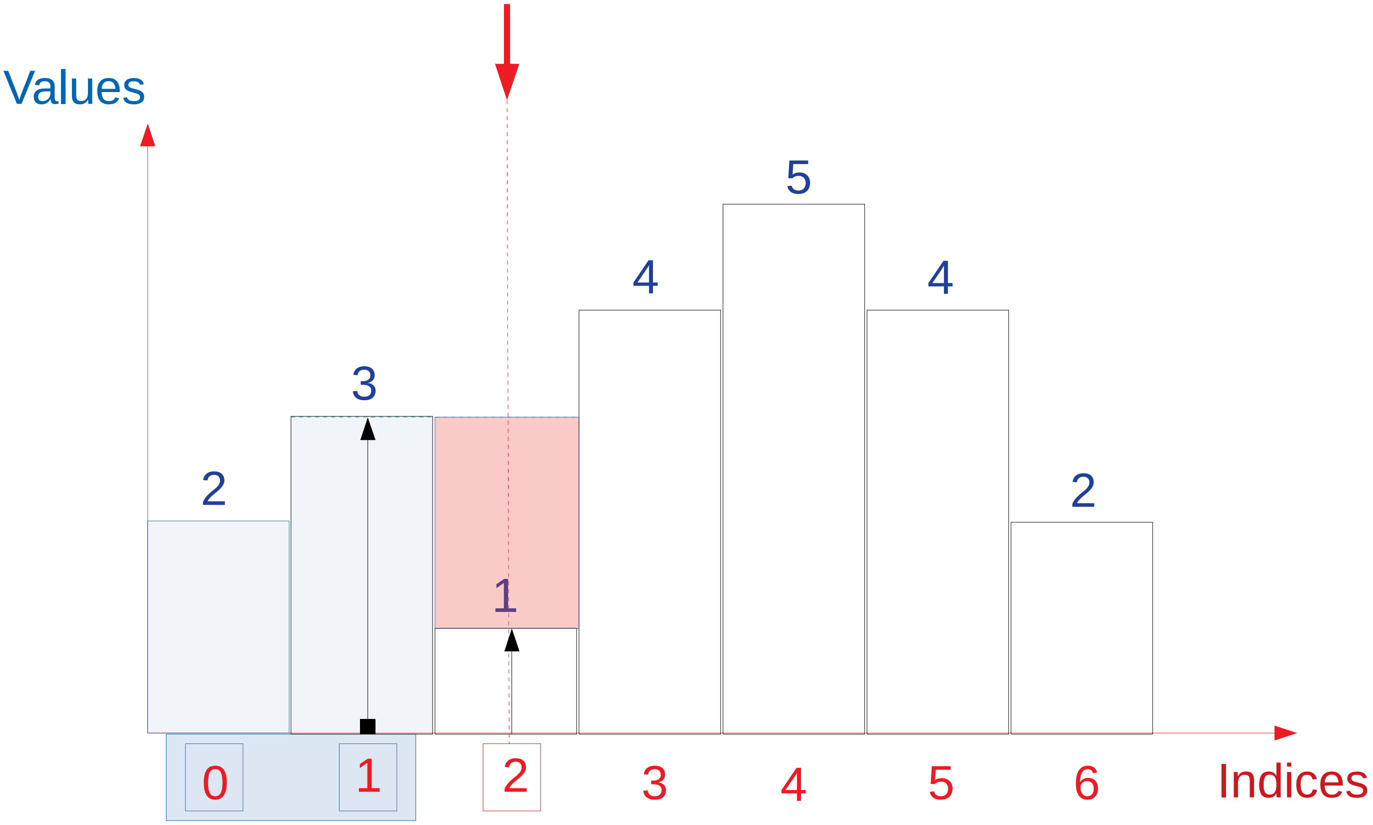
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



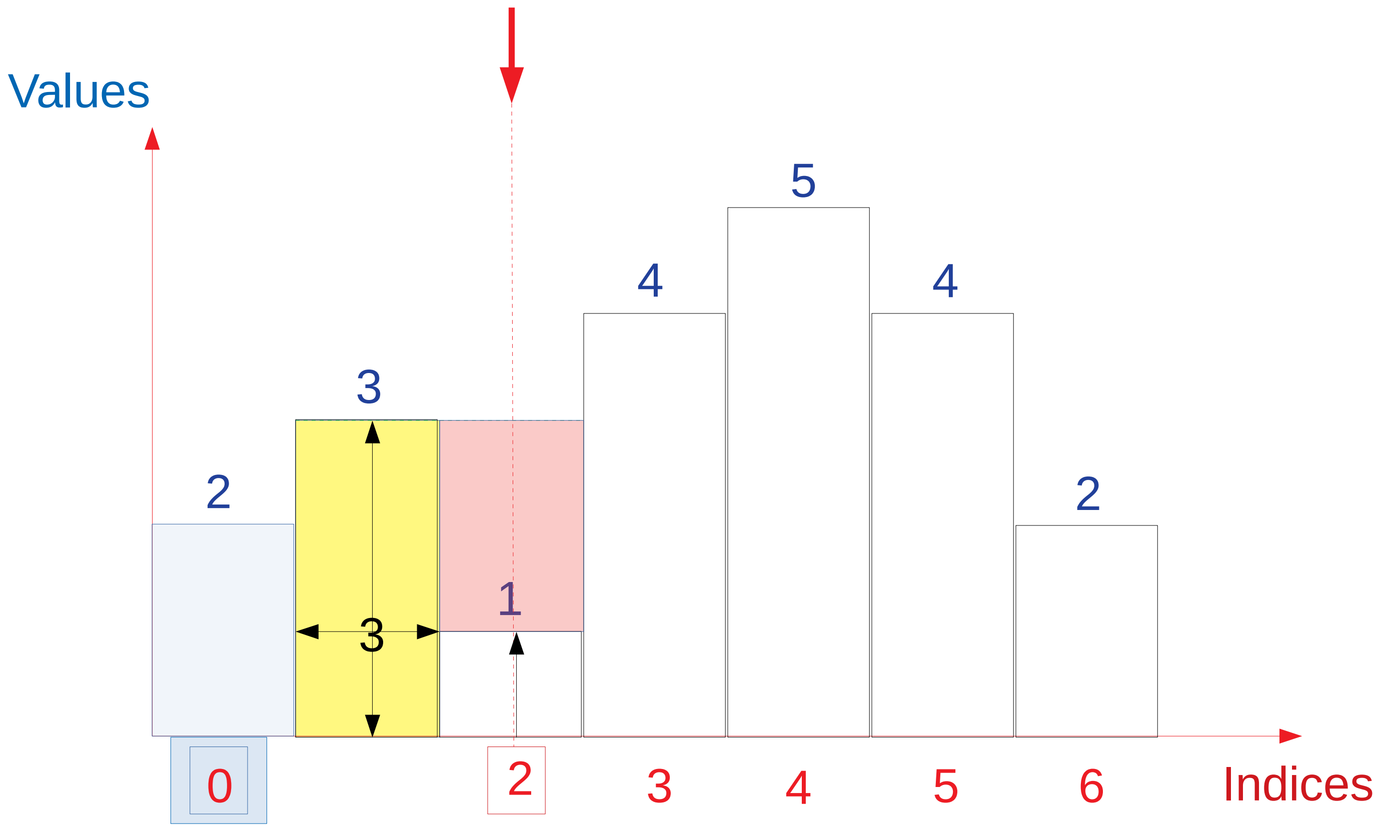
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



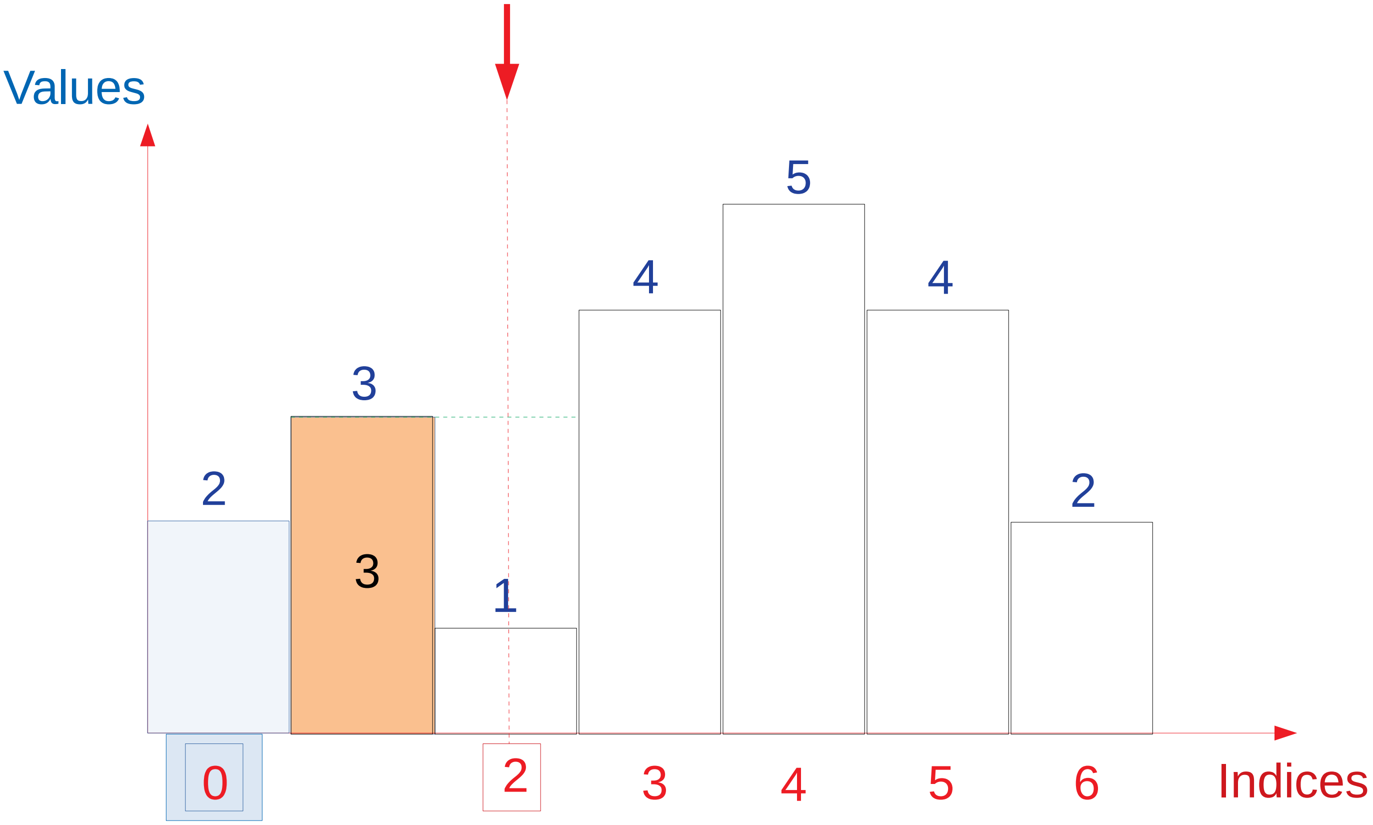
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
    (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                  else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



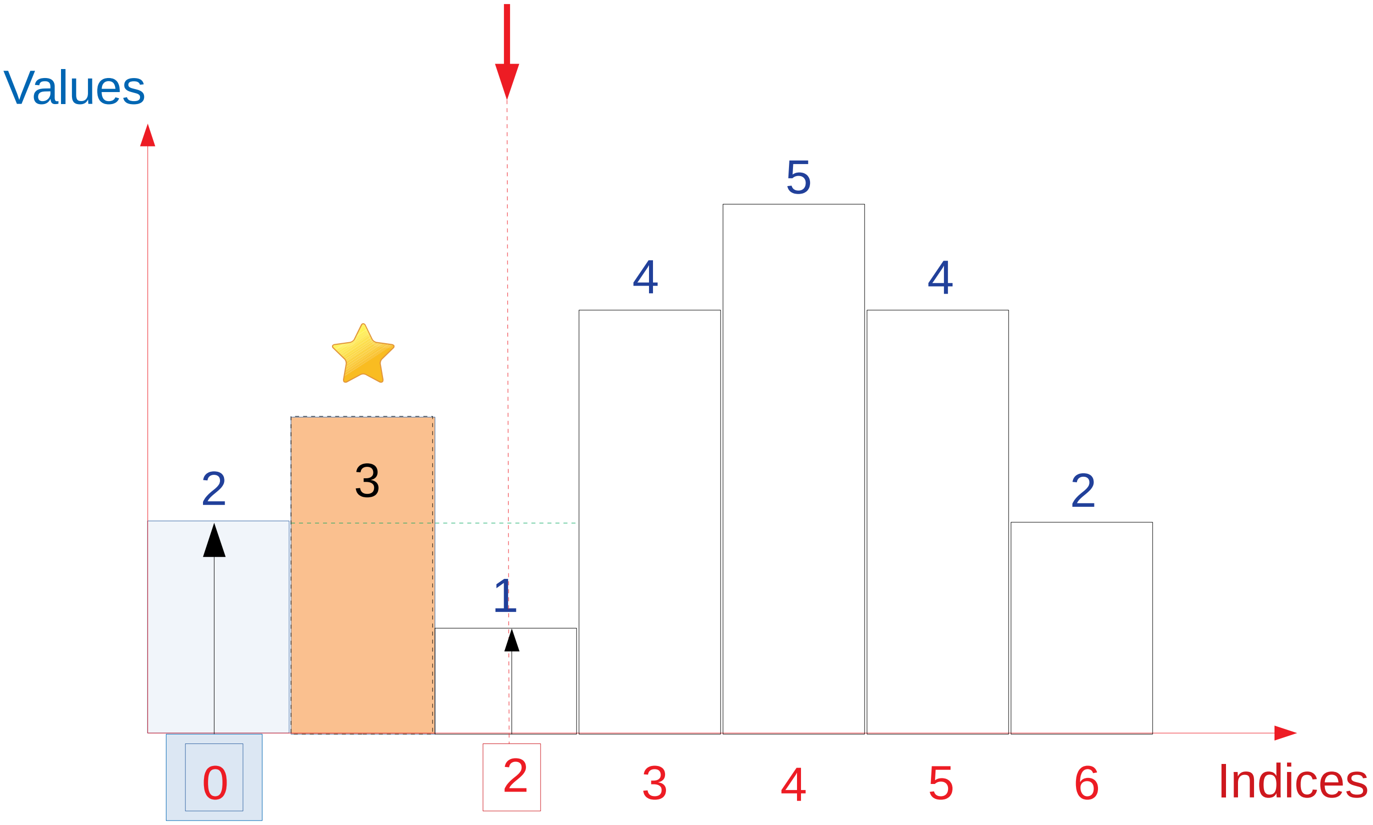
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```

3

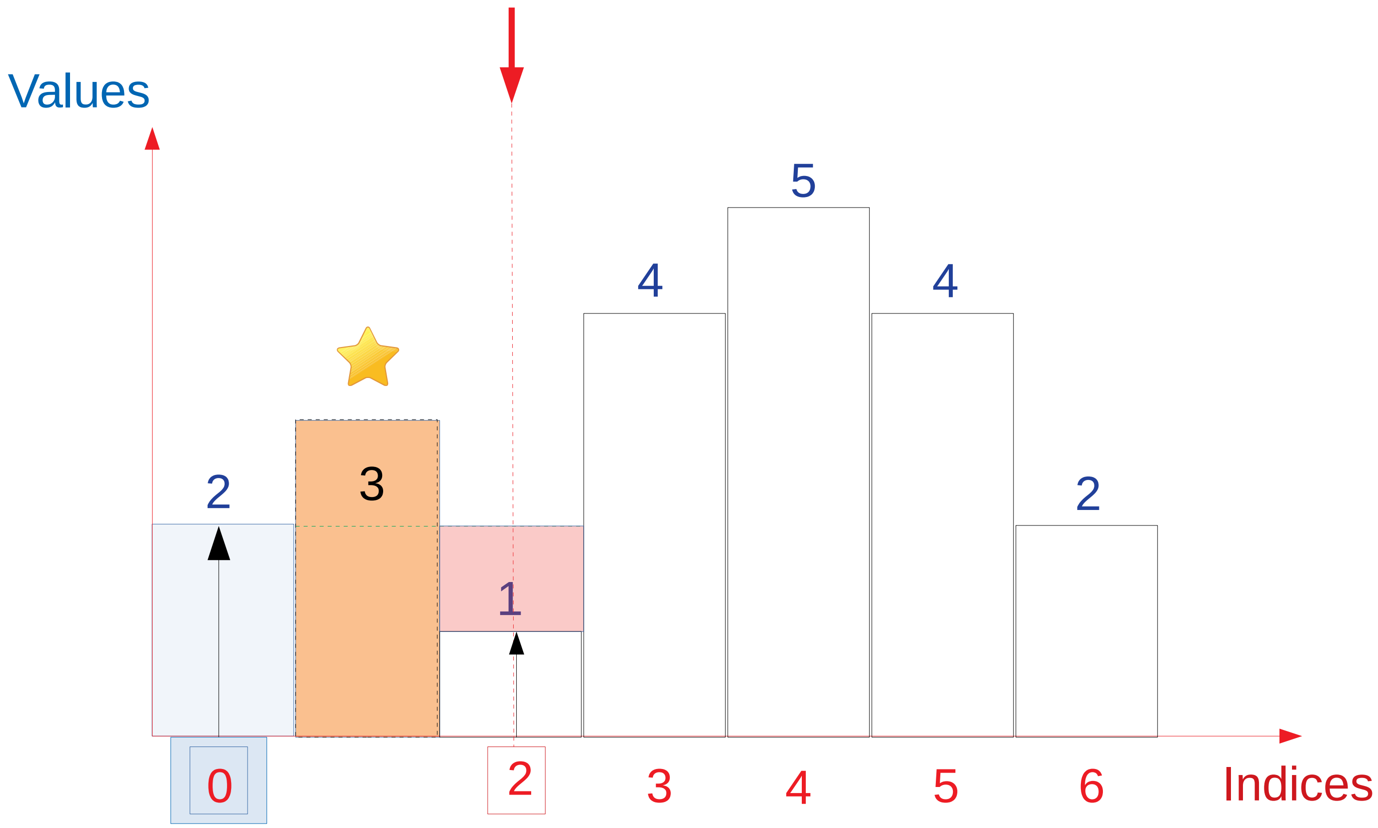
(2-1) – 0 = 1



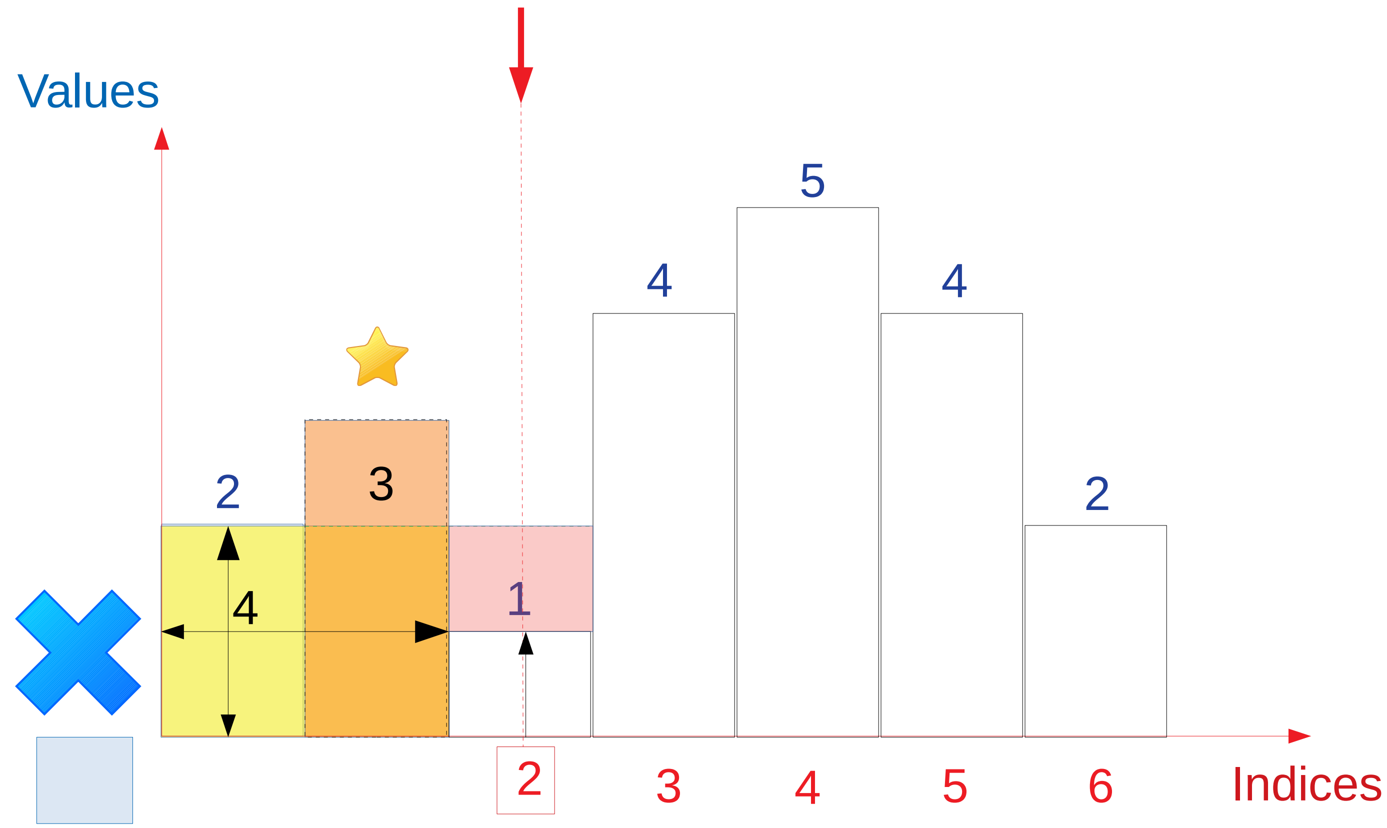
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



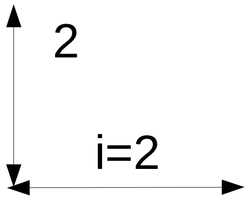
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
     (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                  else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```

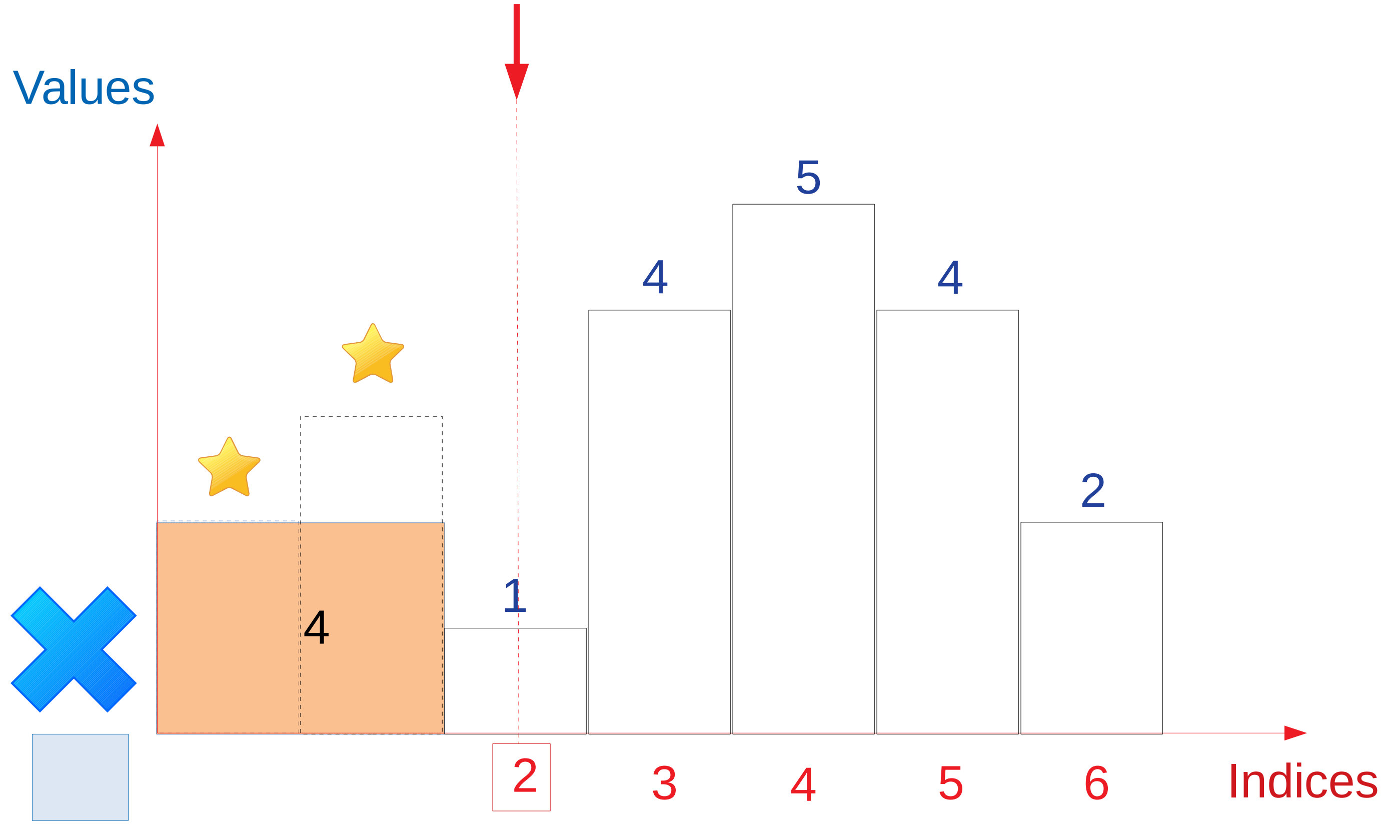


```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```

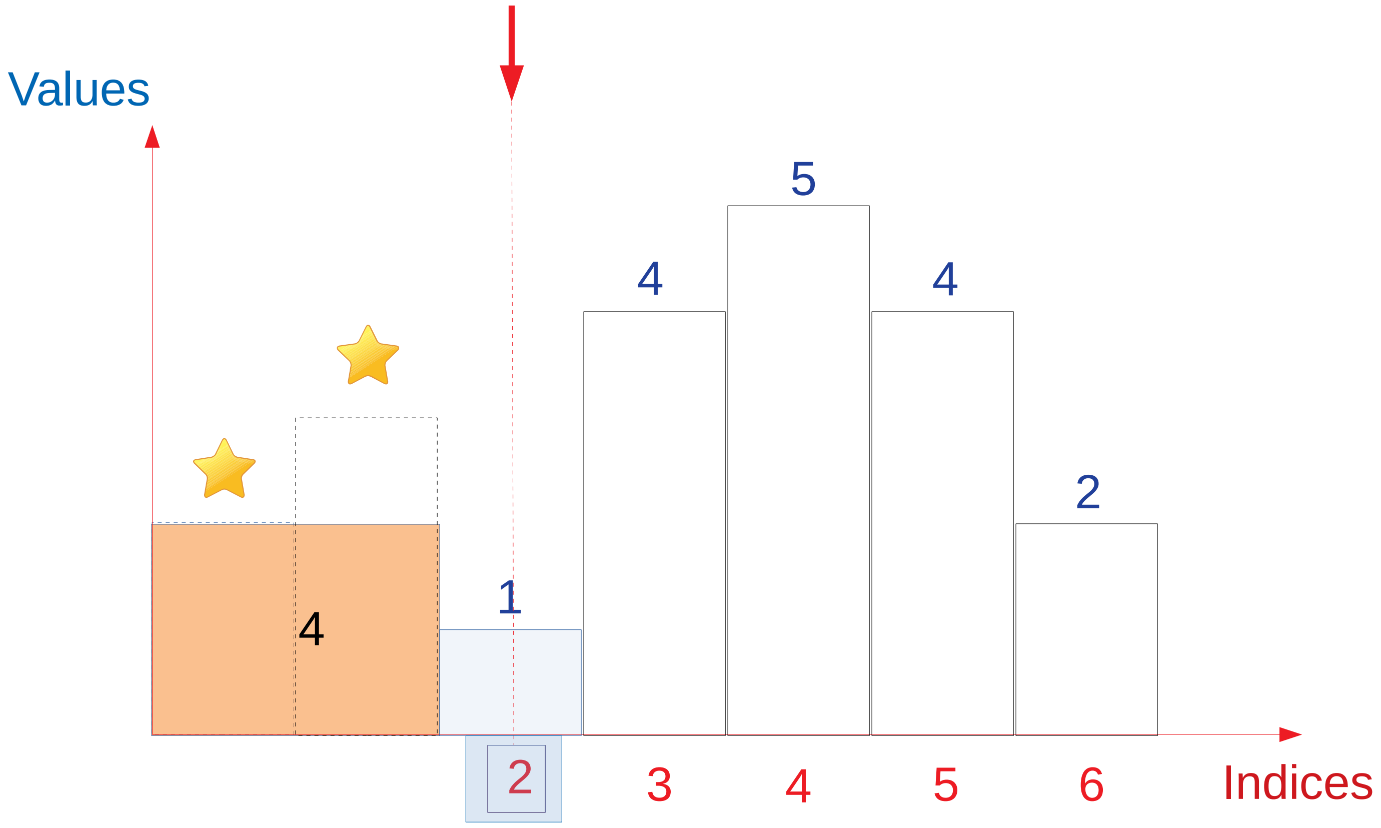


```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```

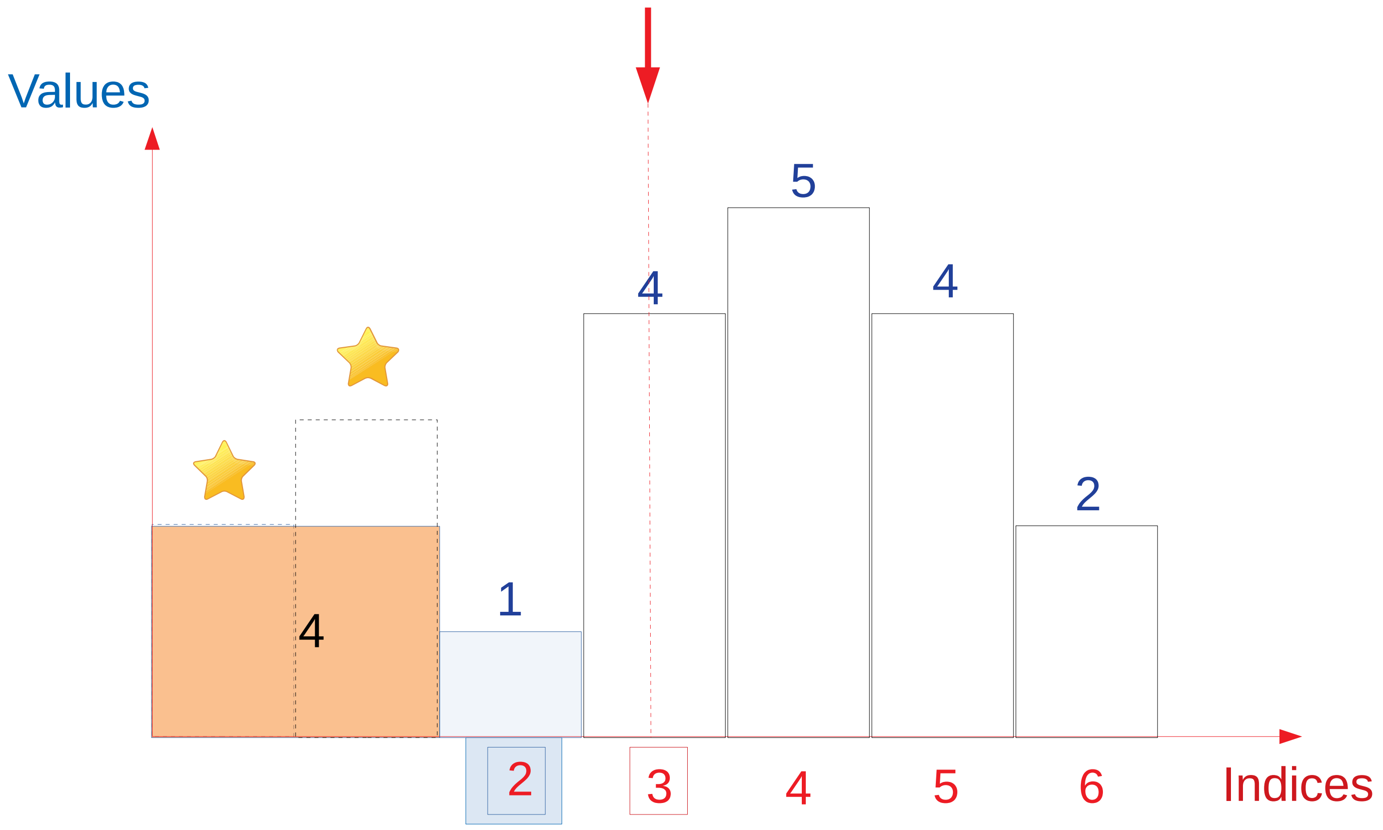




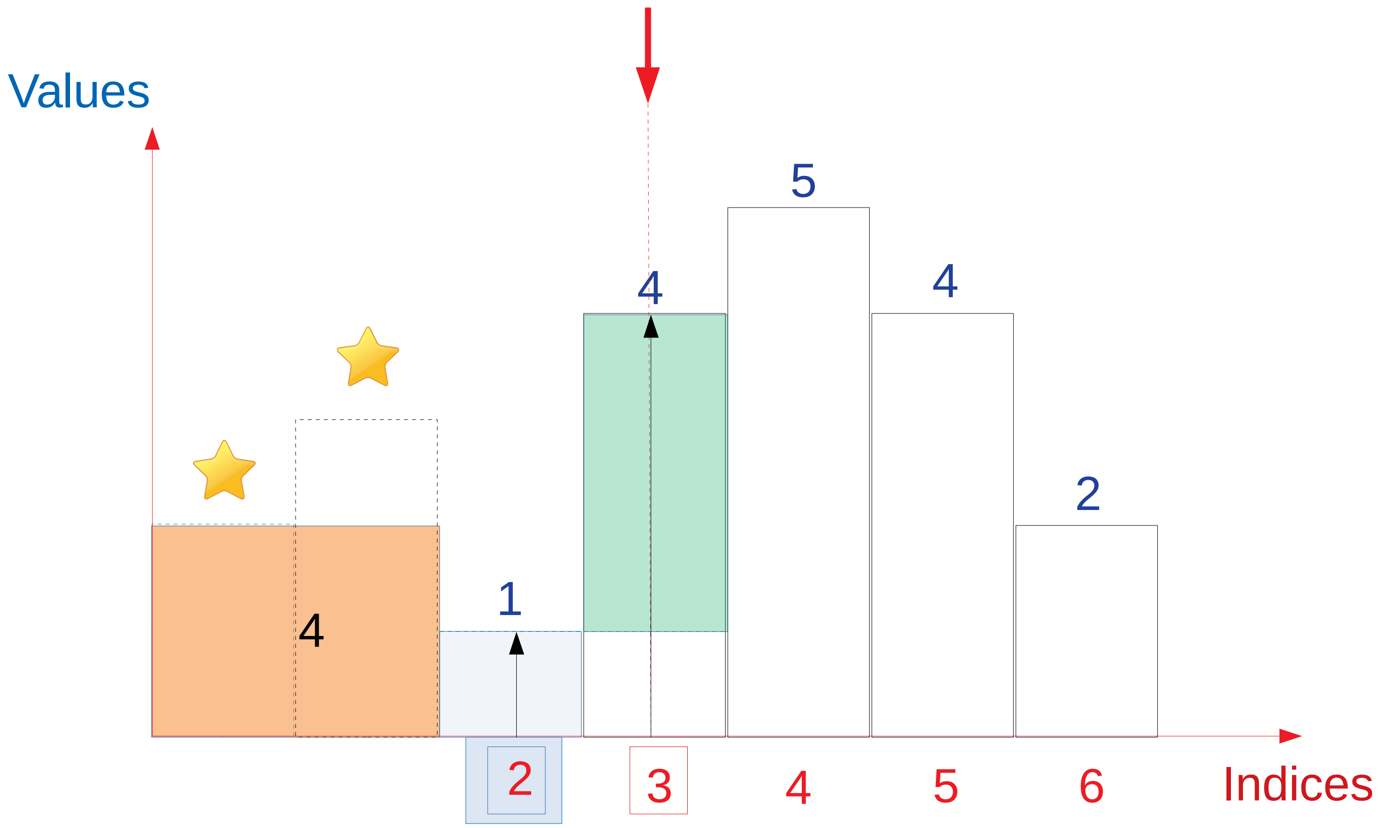
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



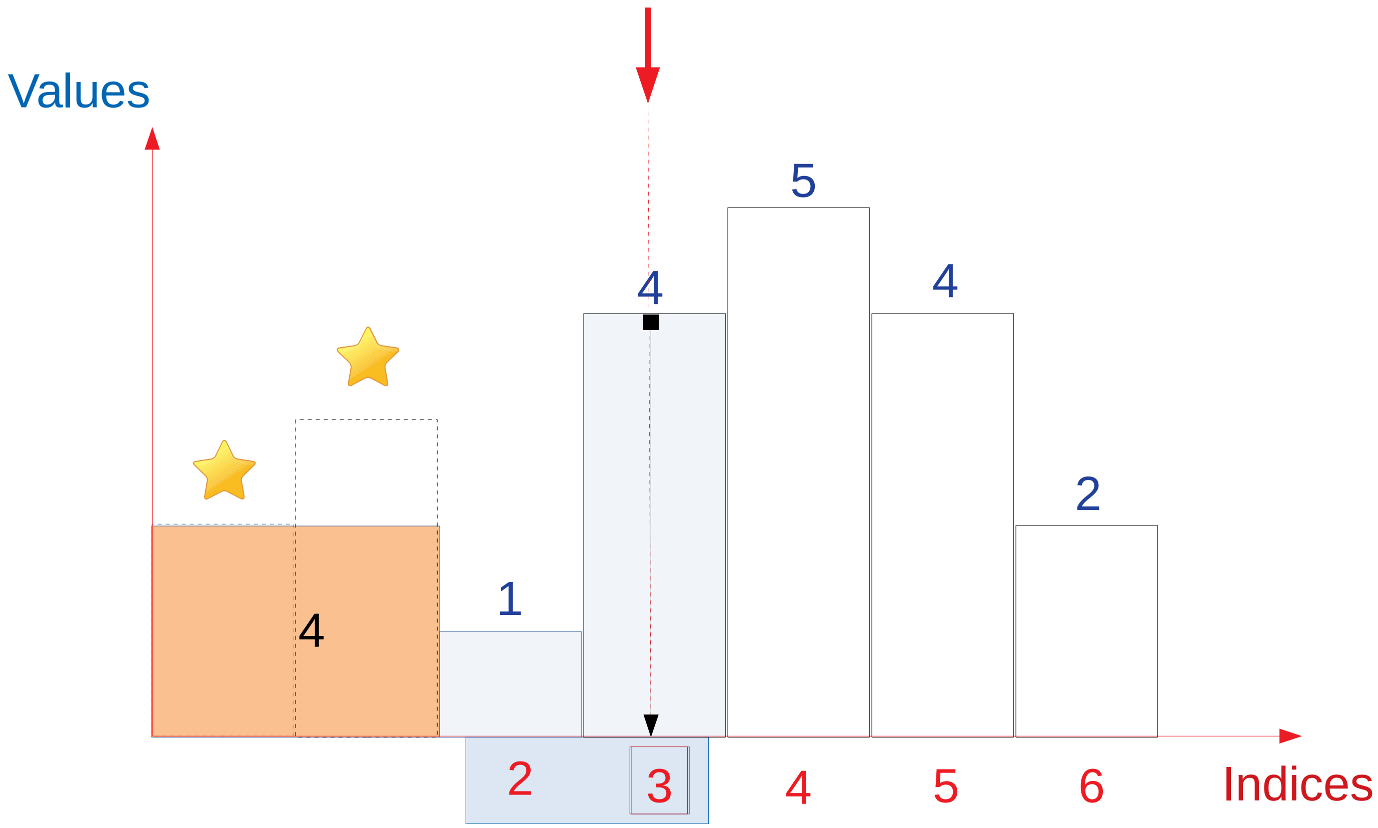
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



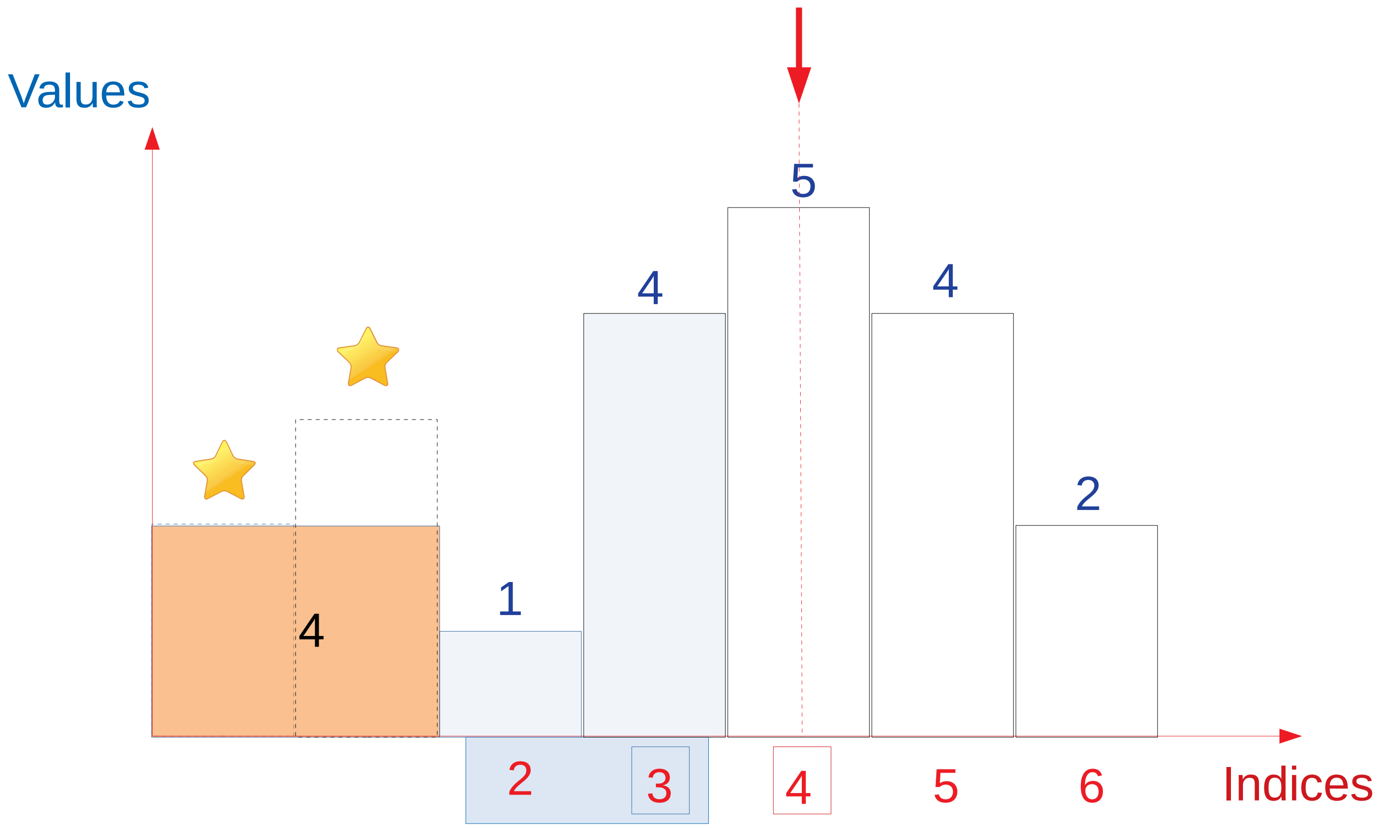
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



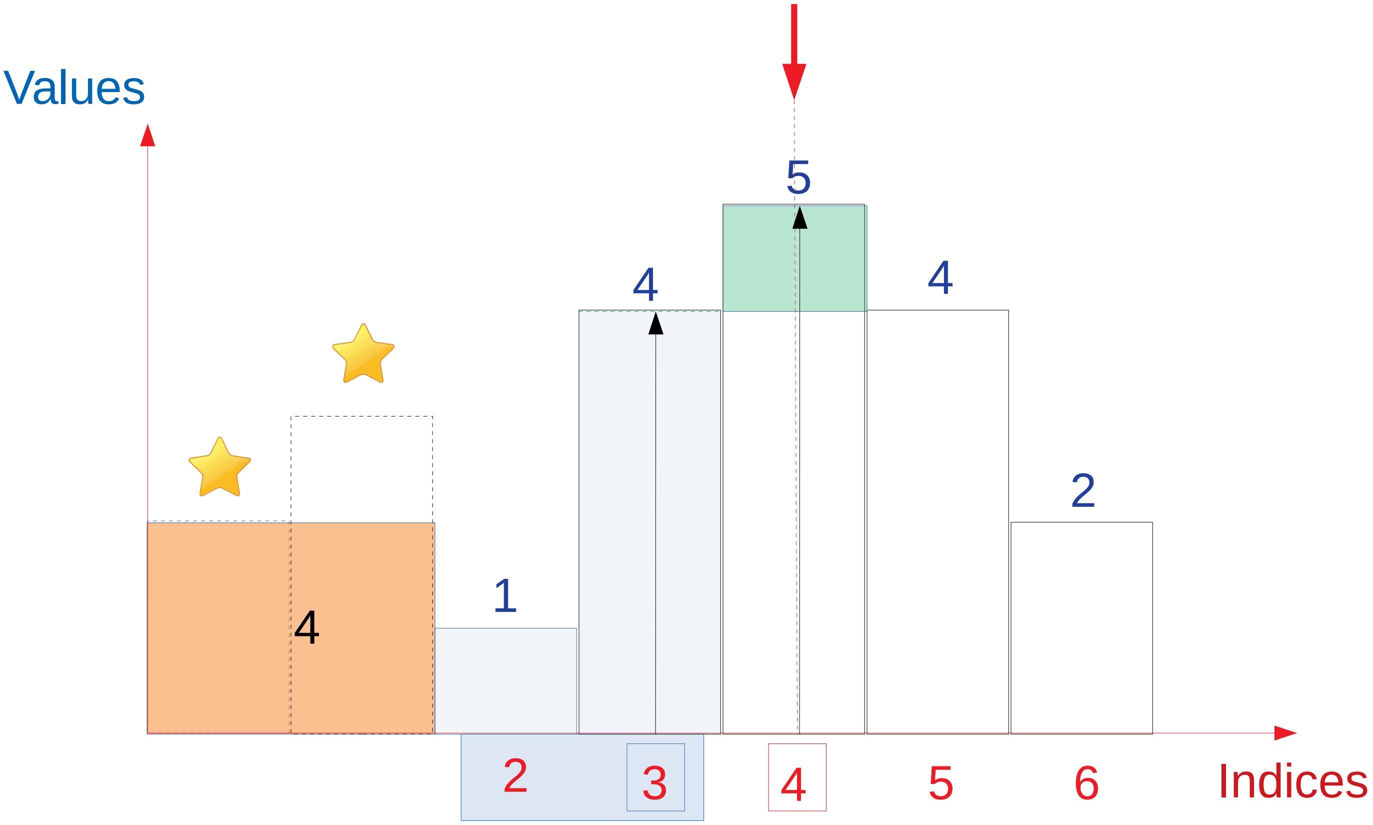
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



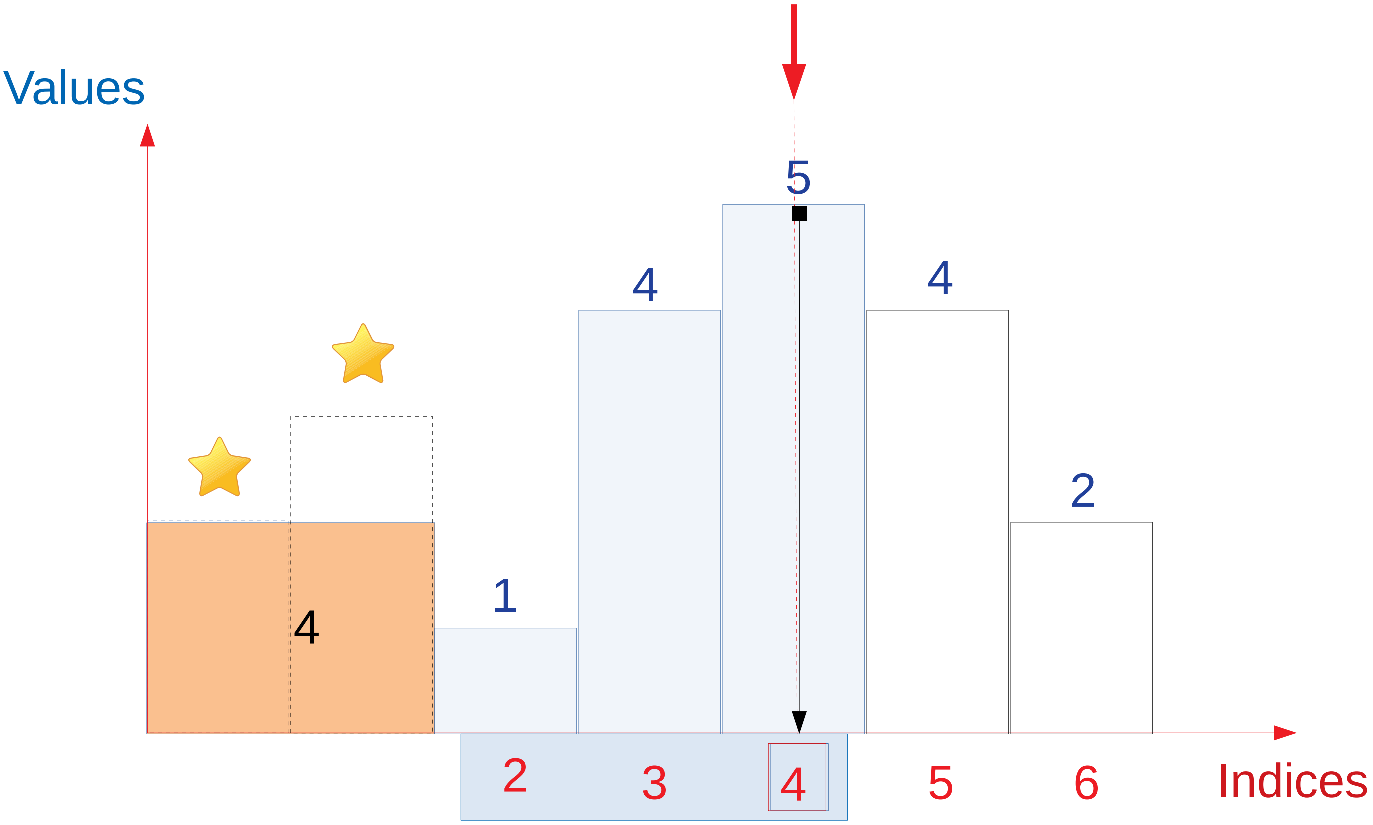
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
        (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                  else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



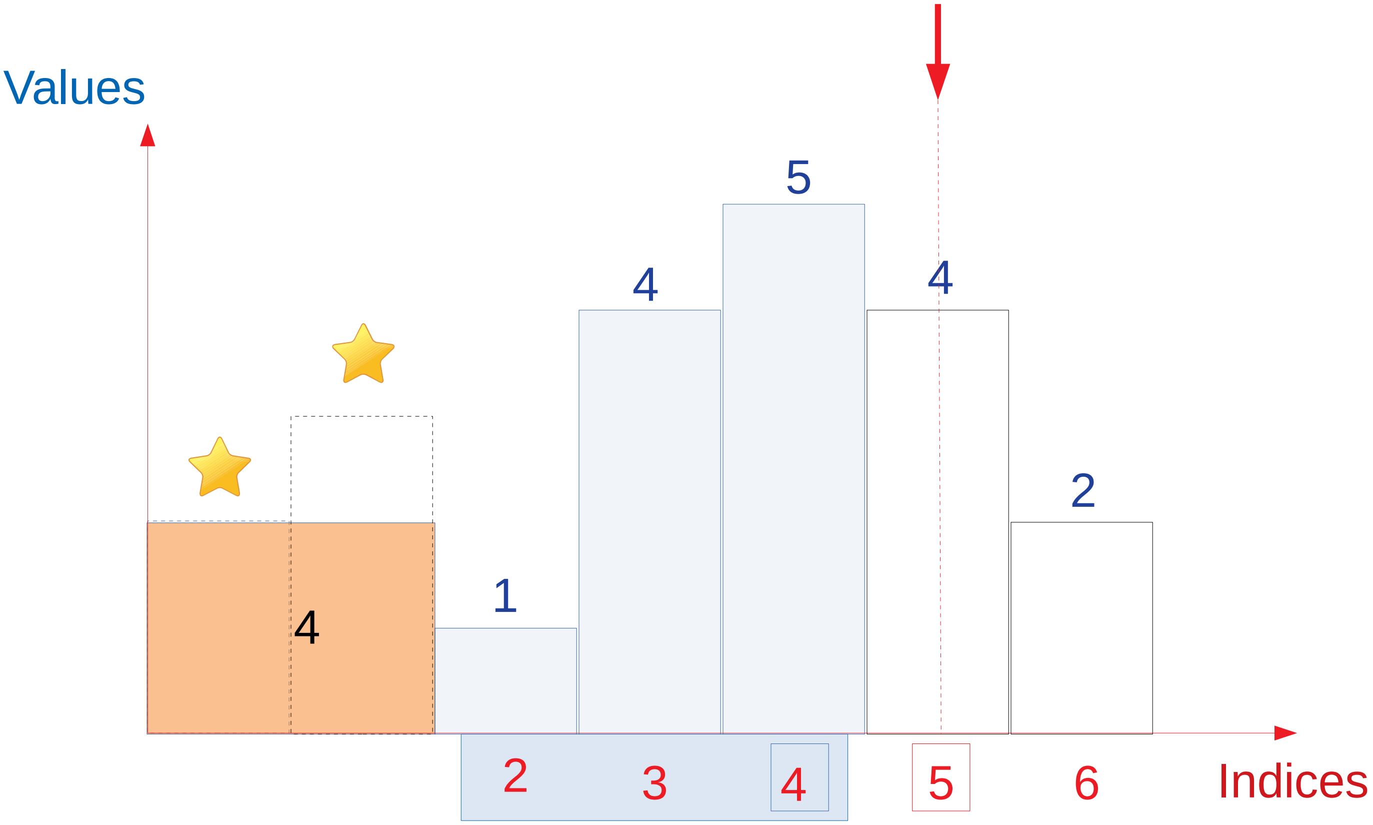
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
        (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                  else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



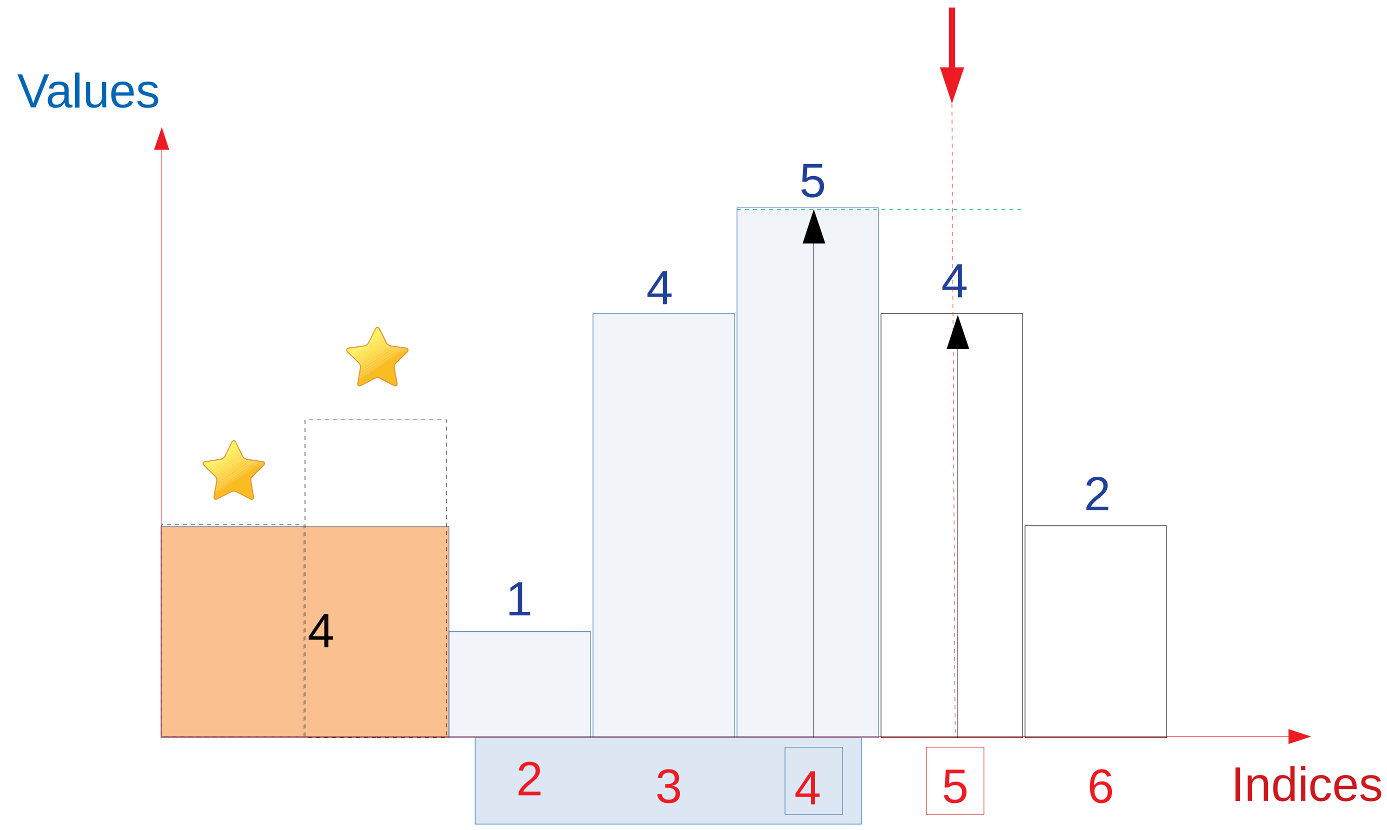
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



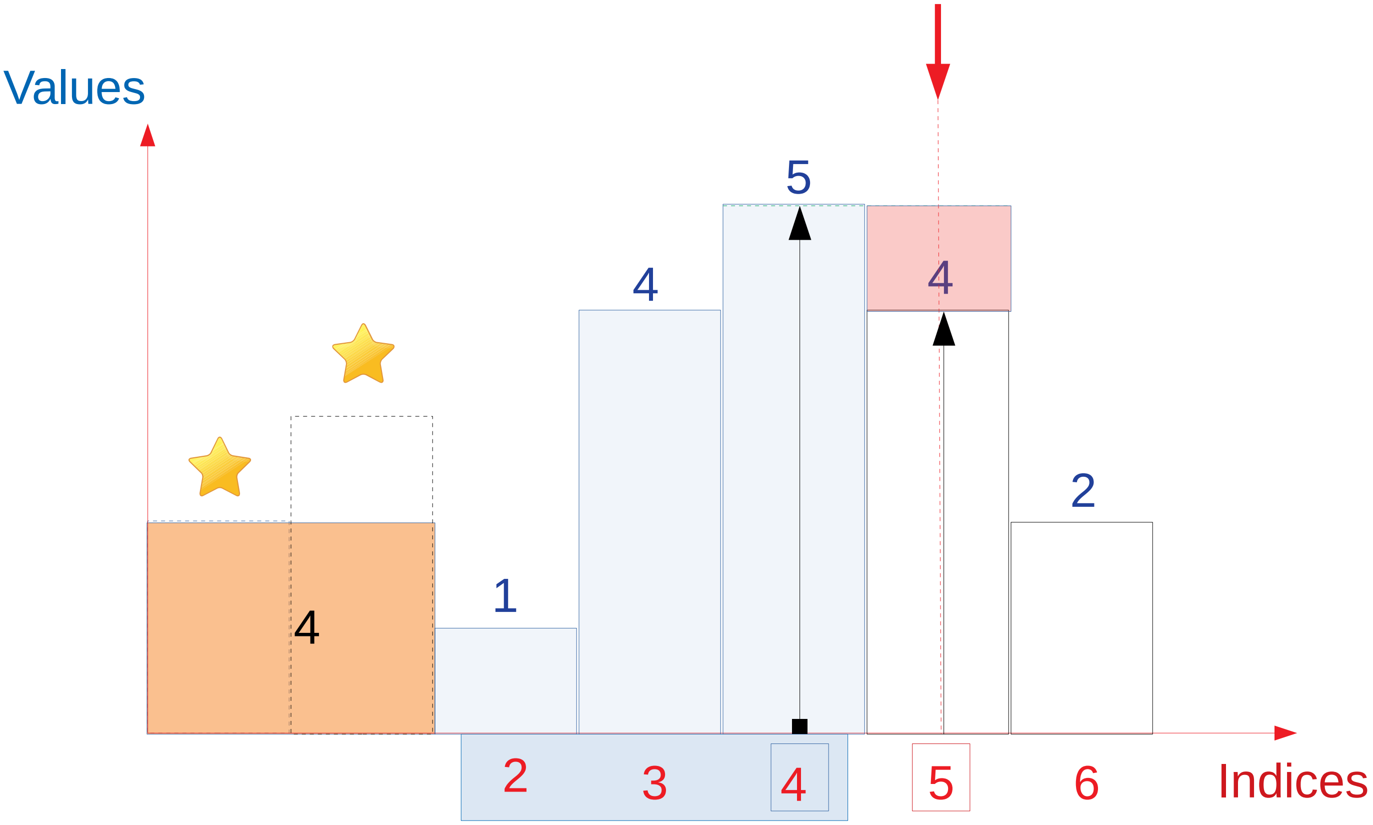
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



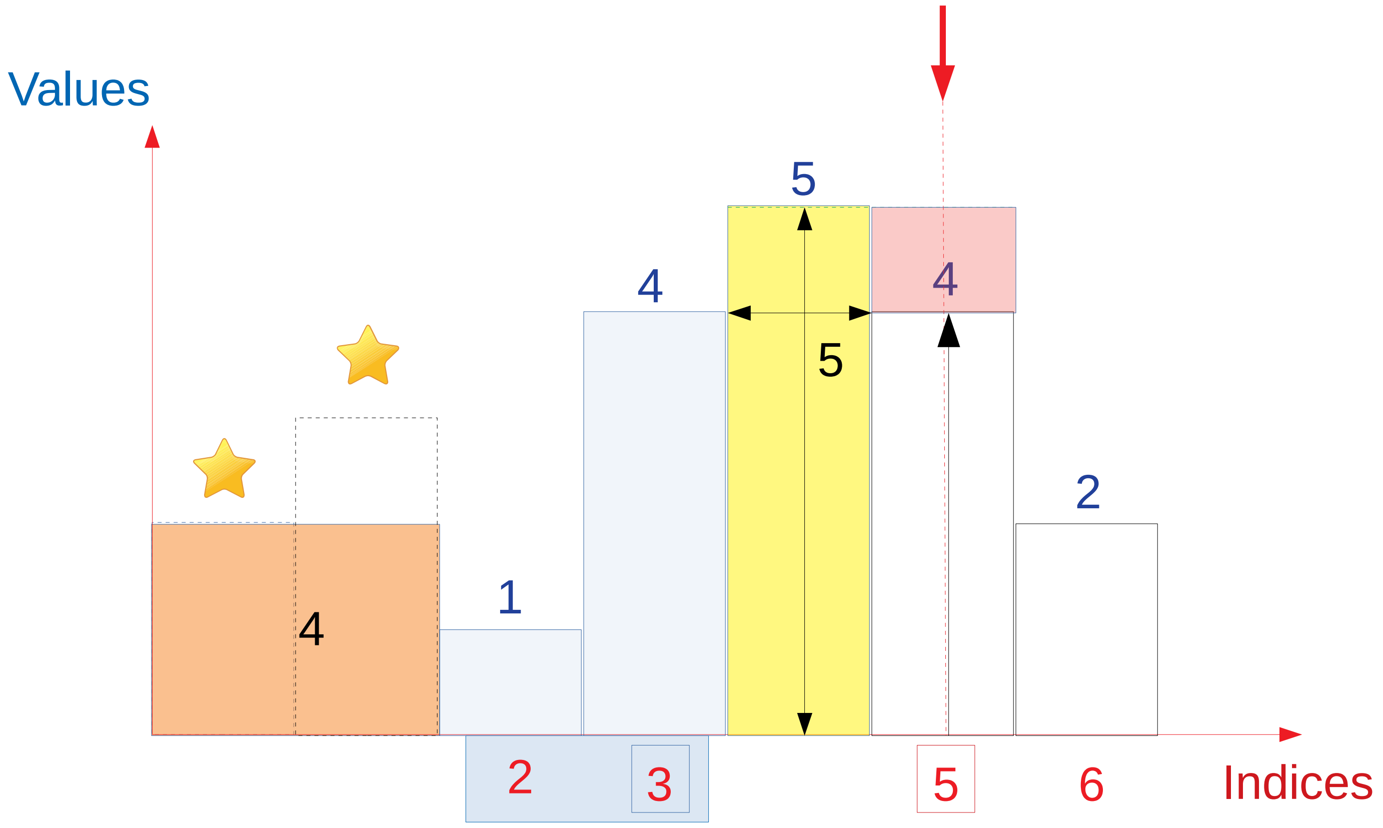
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
    (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                  else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



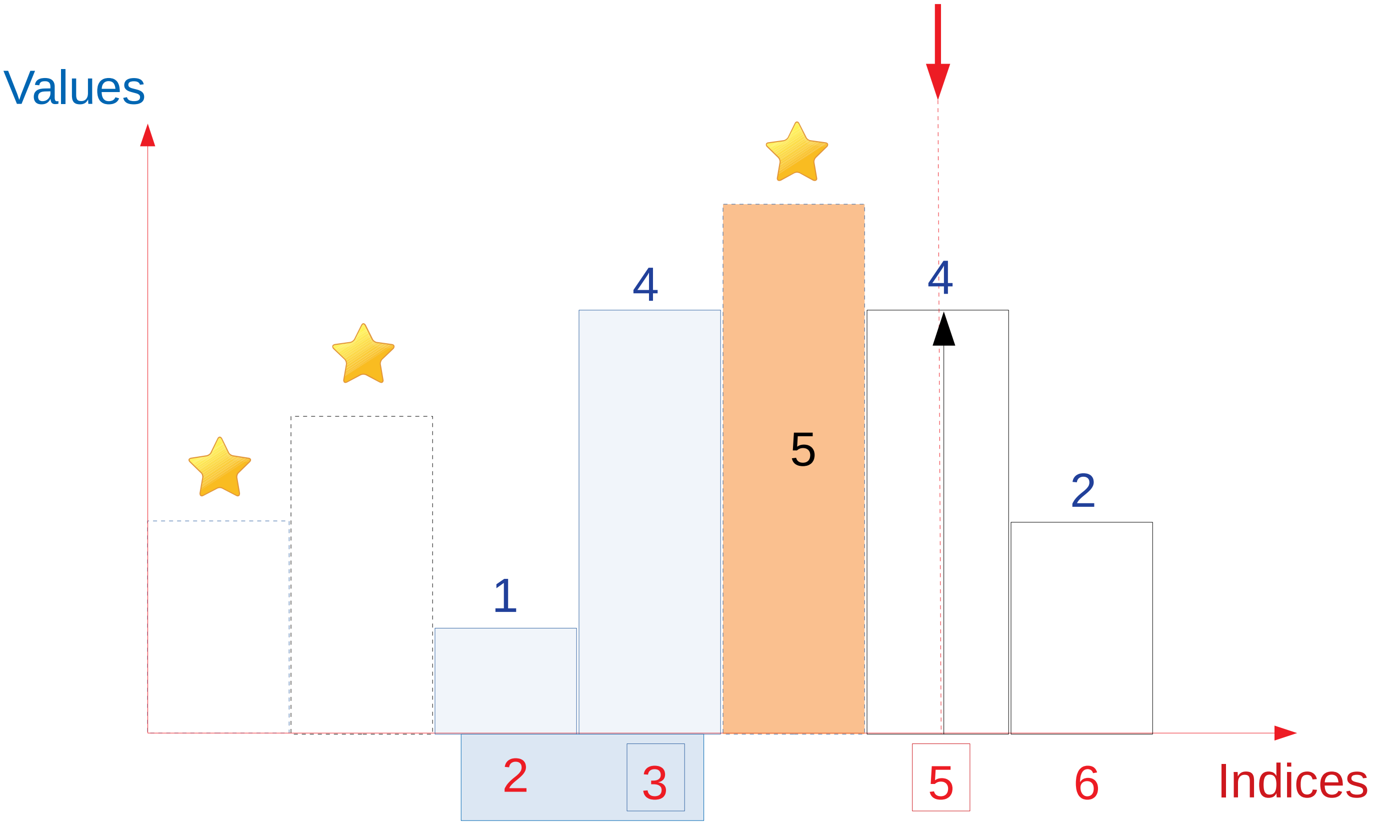
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                  else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



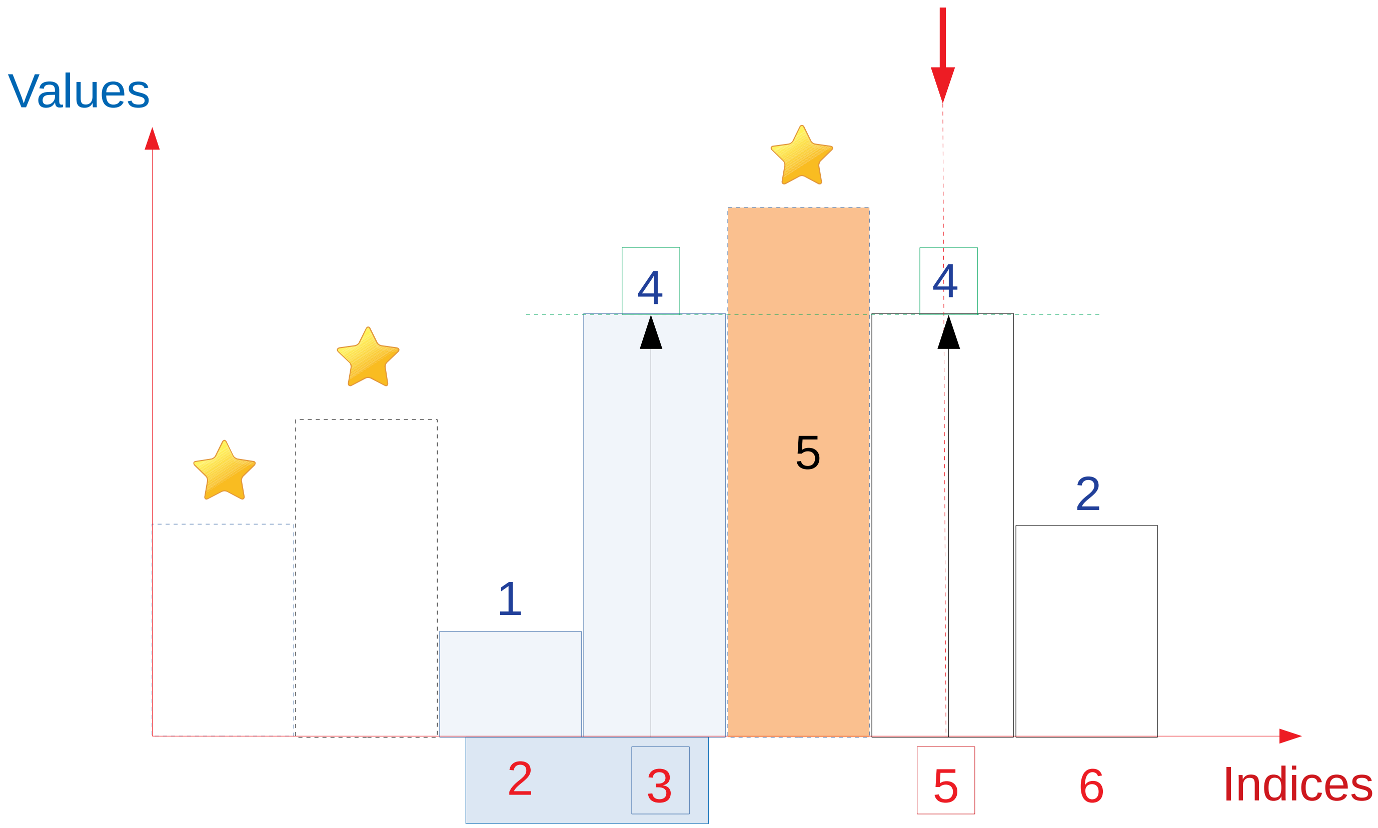
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
      (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```

5

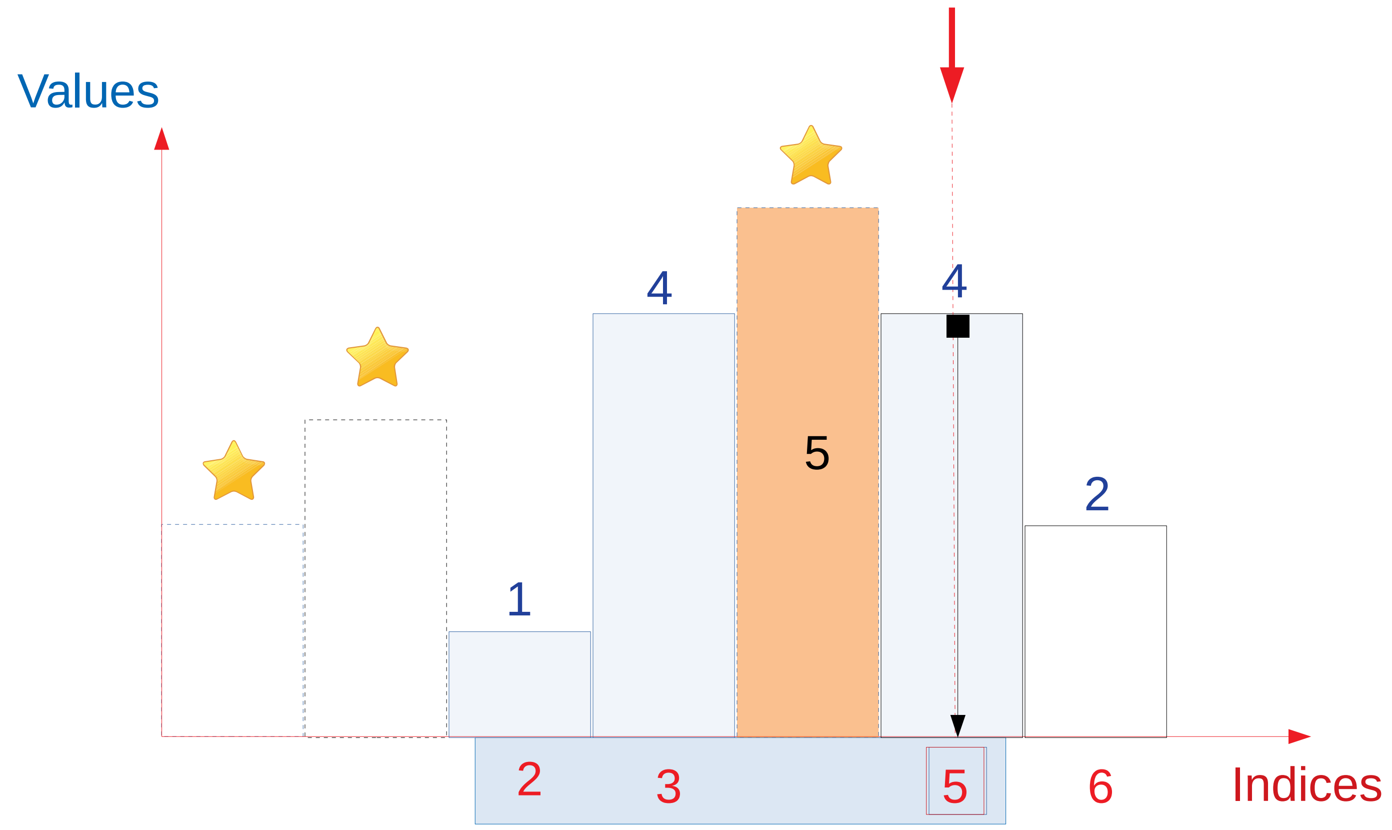
(5-1) – 3 = 1



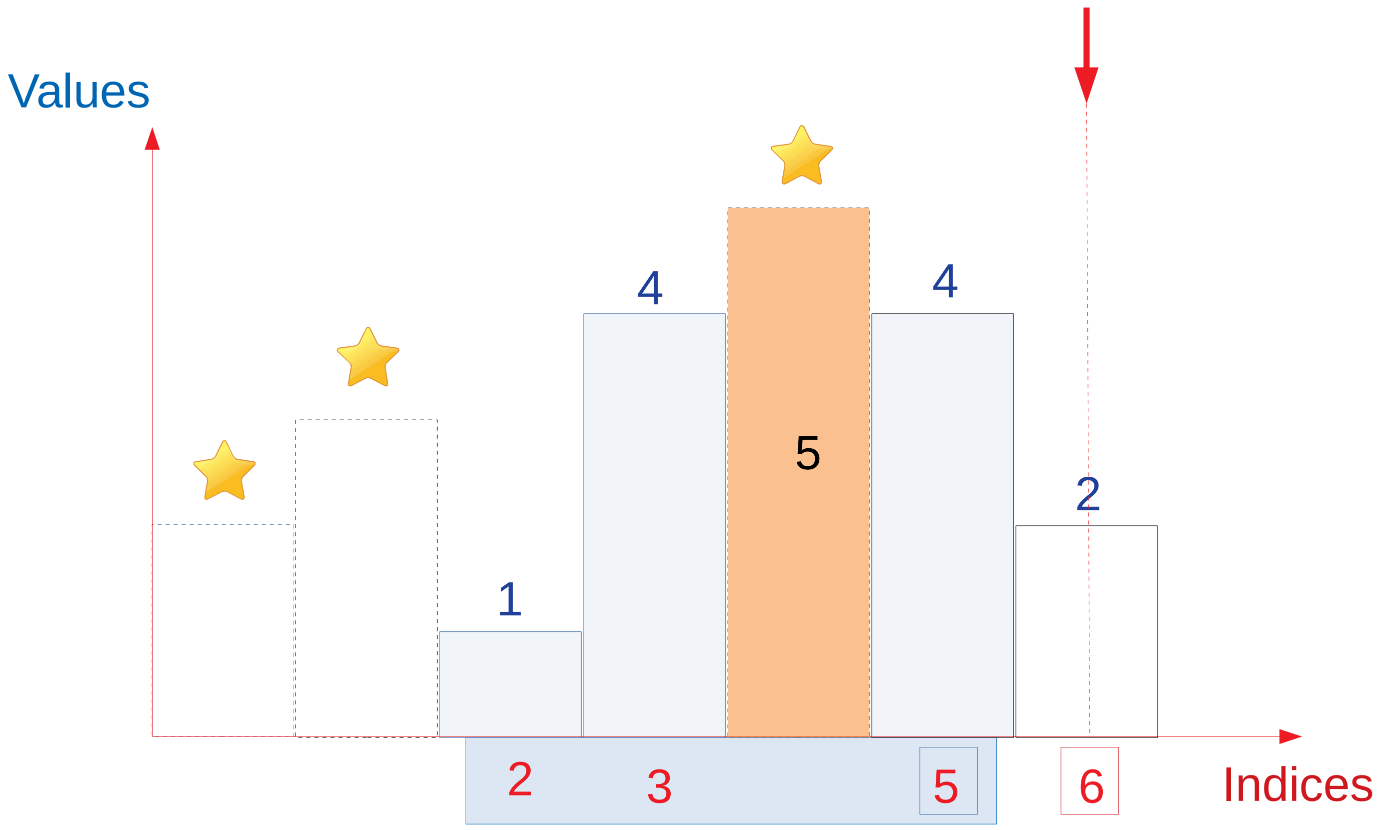
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
    (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
             (i if isEmpty(s)
                else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



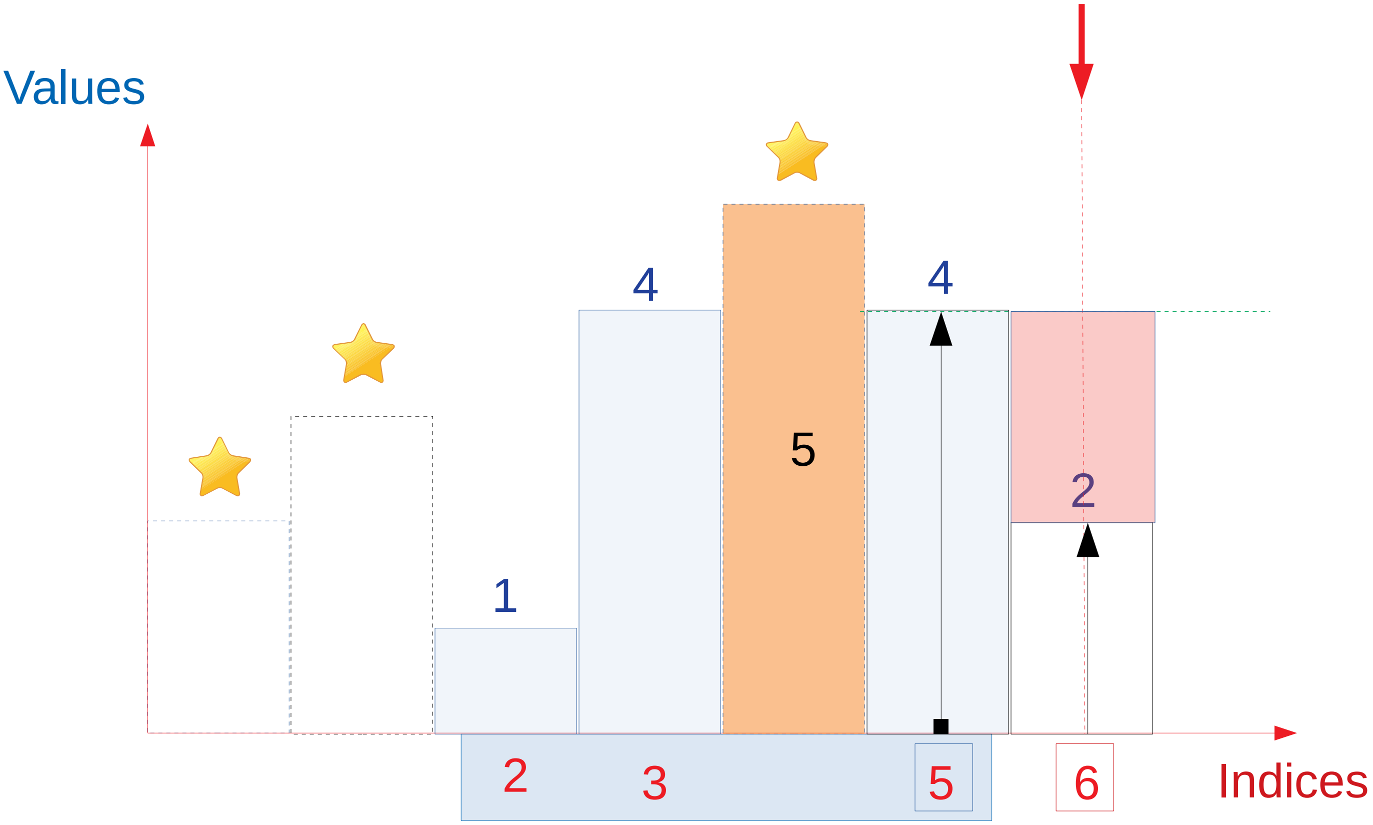
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



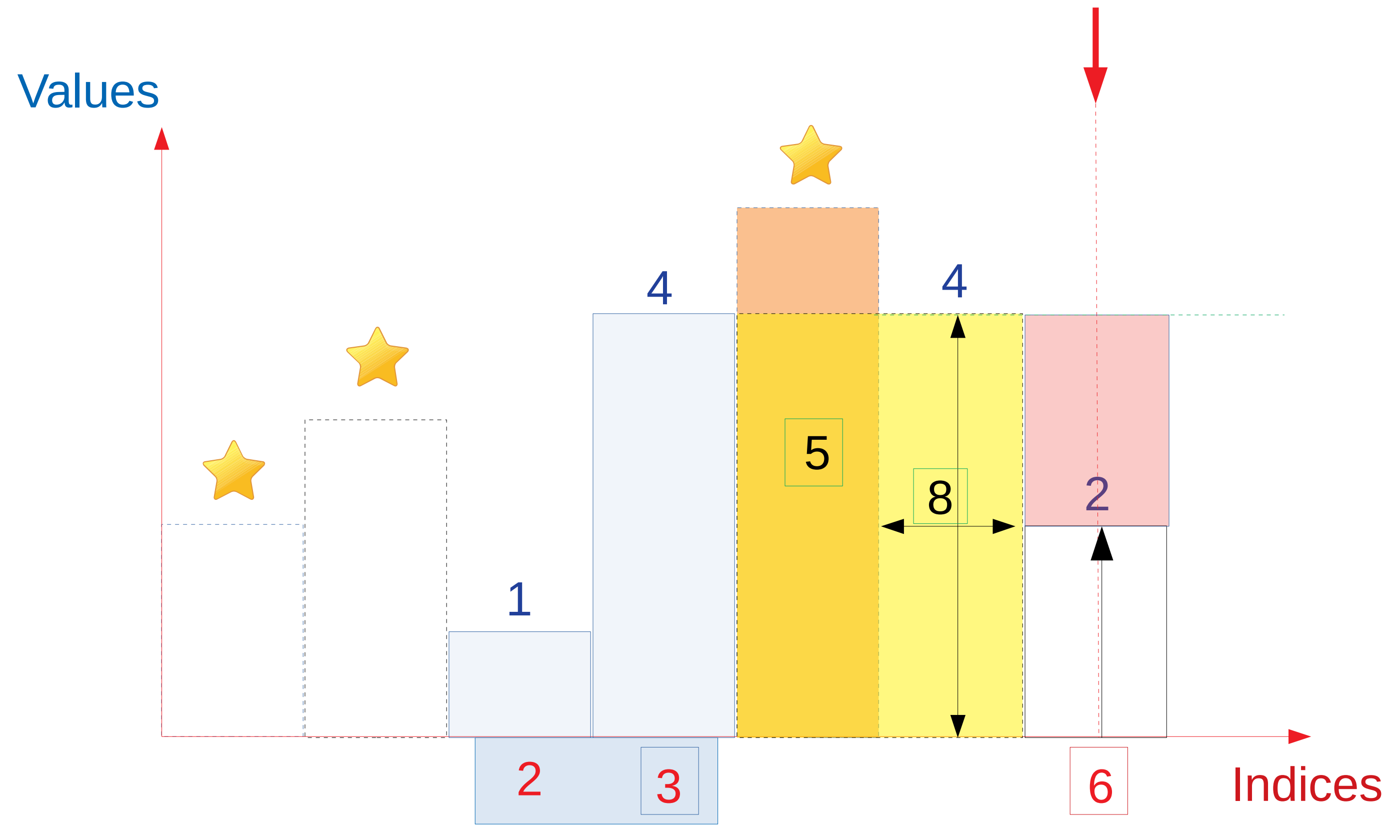
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



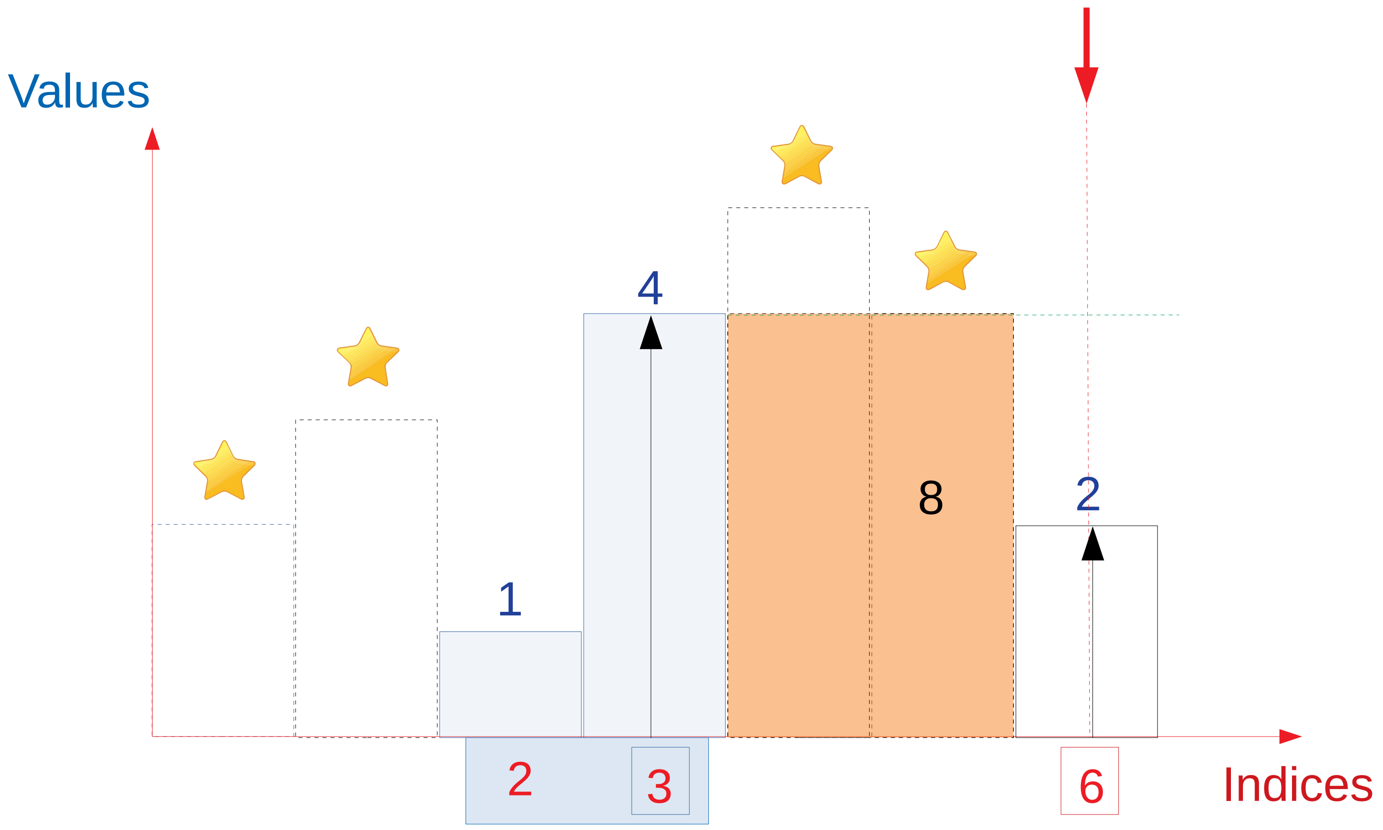
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



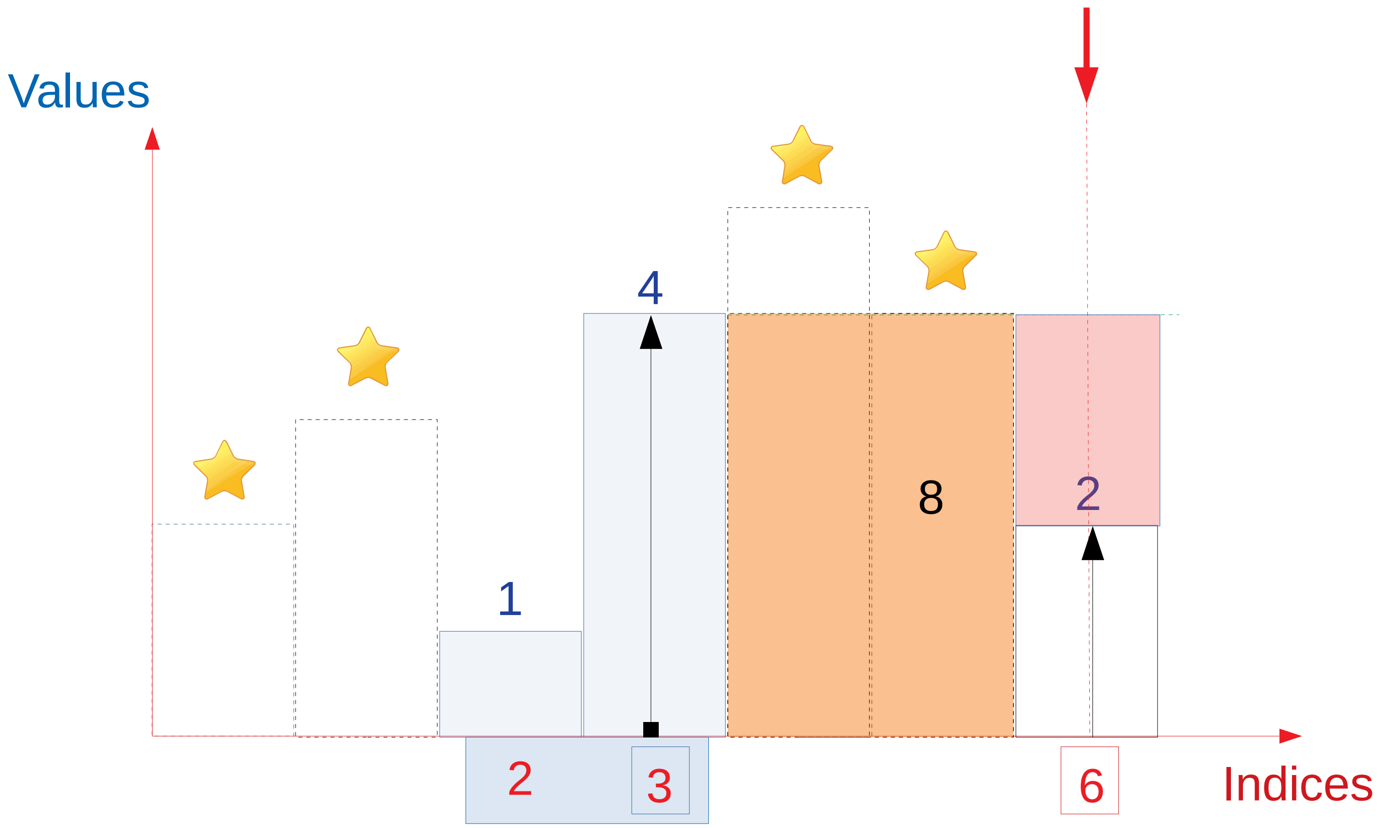
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```

4

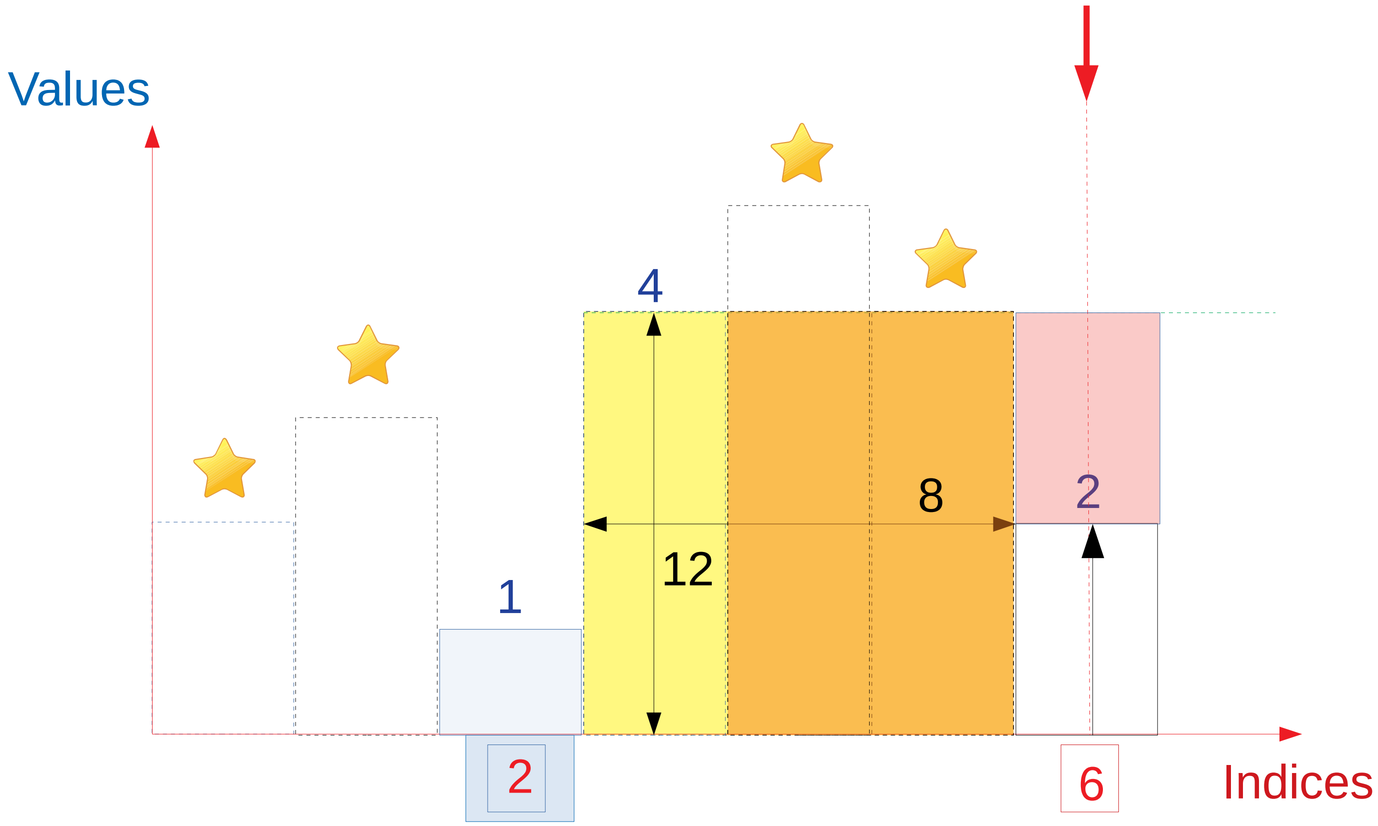
(6-1) – 3 = 2



```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



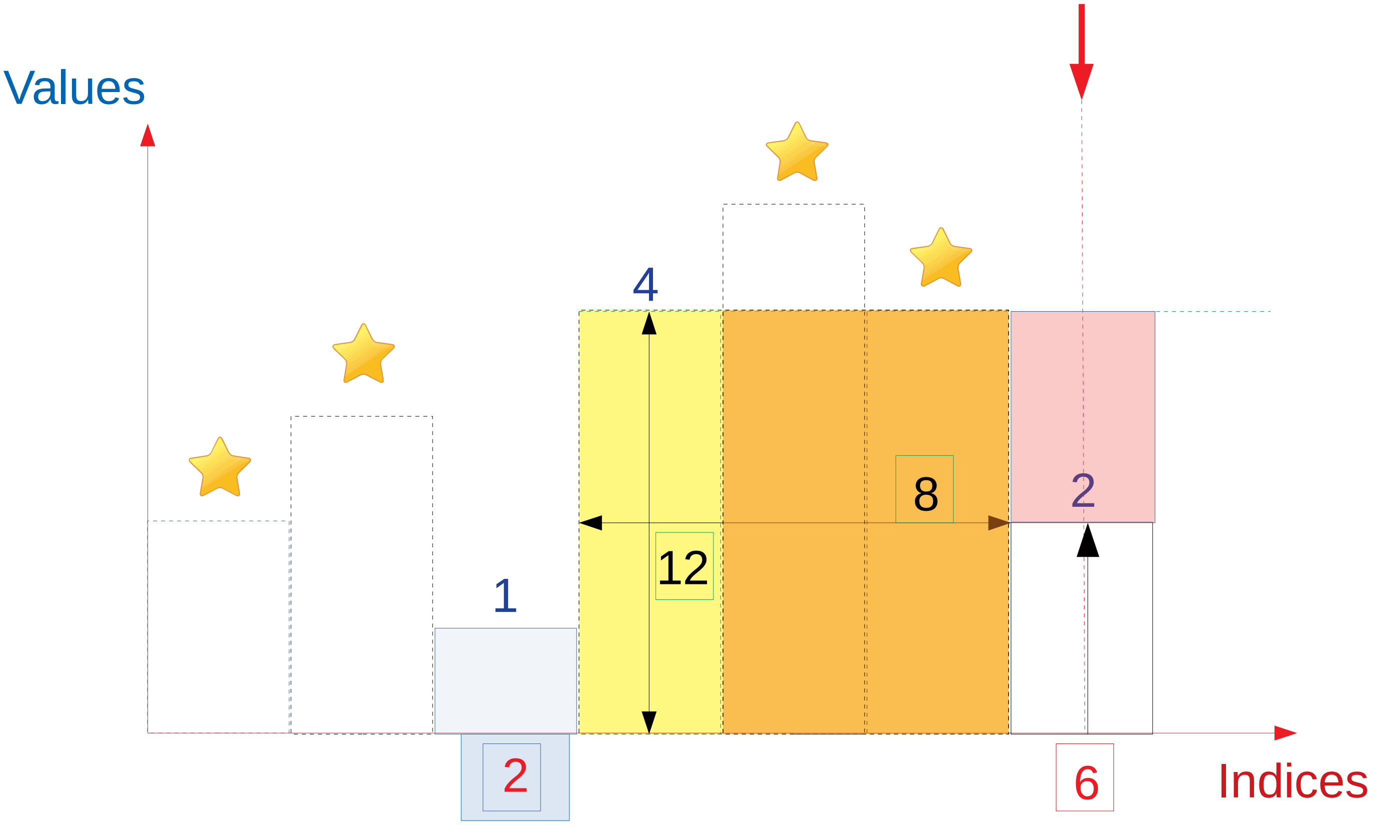
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
             (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



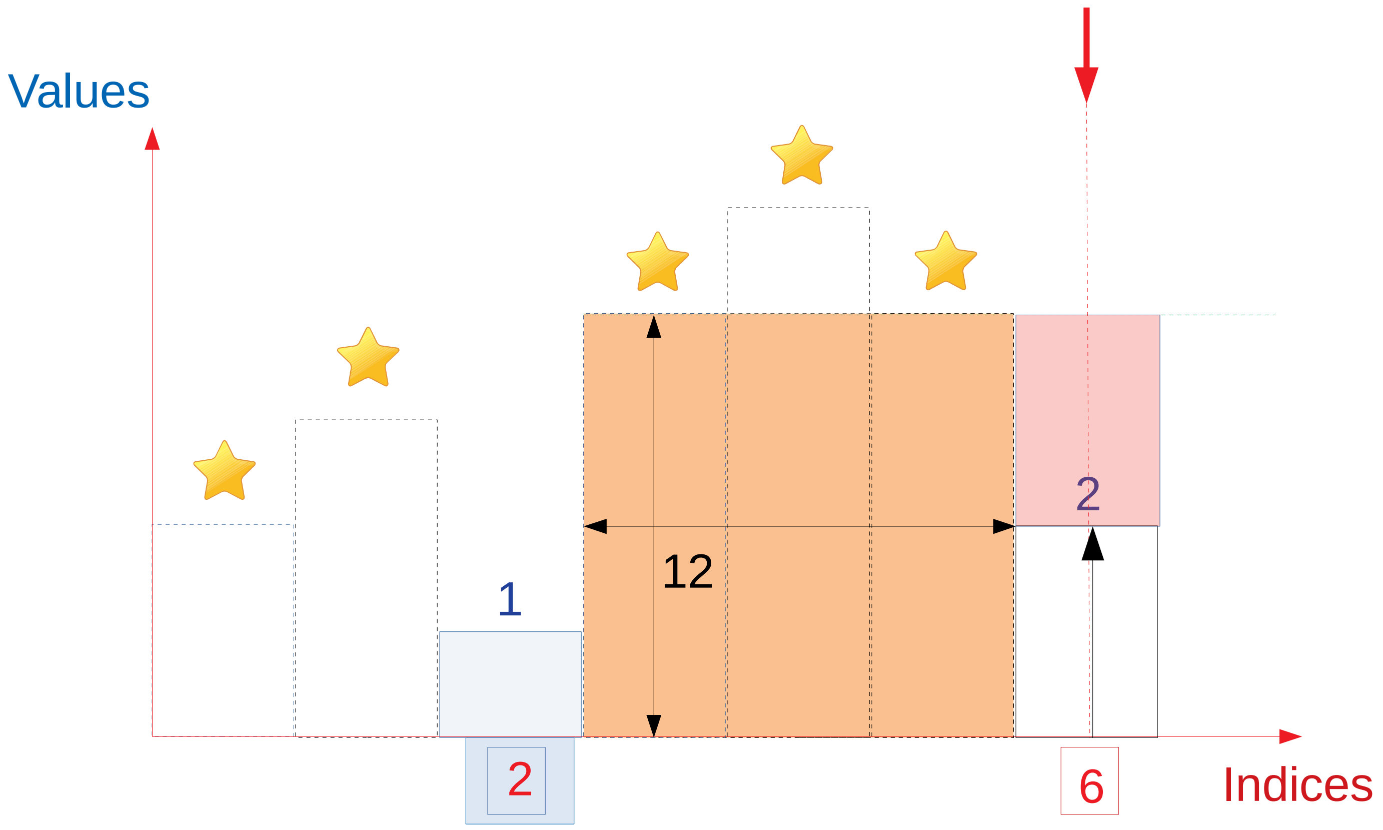
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
             (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```

4

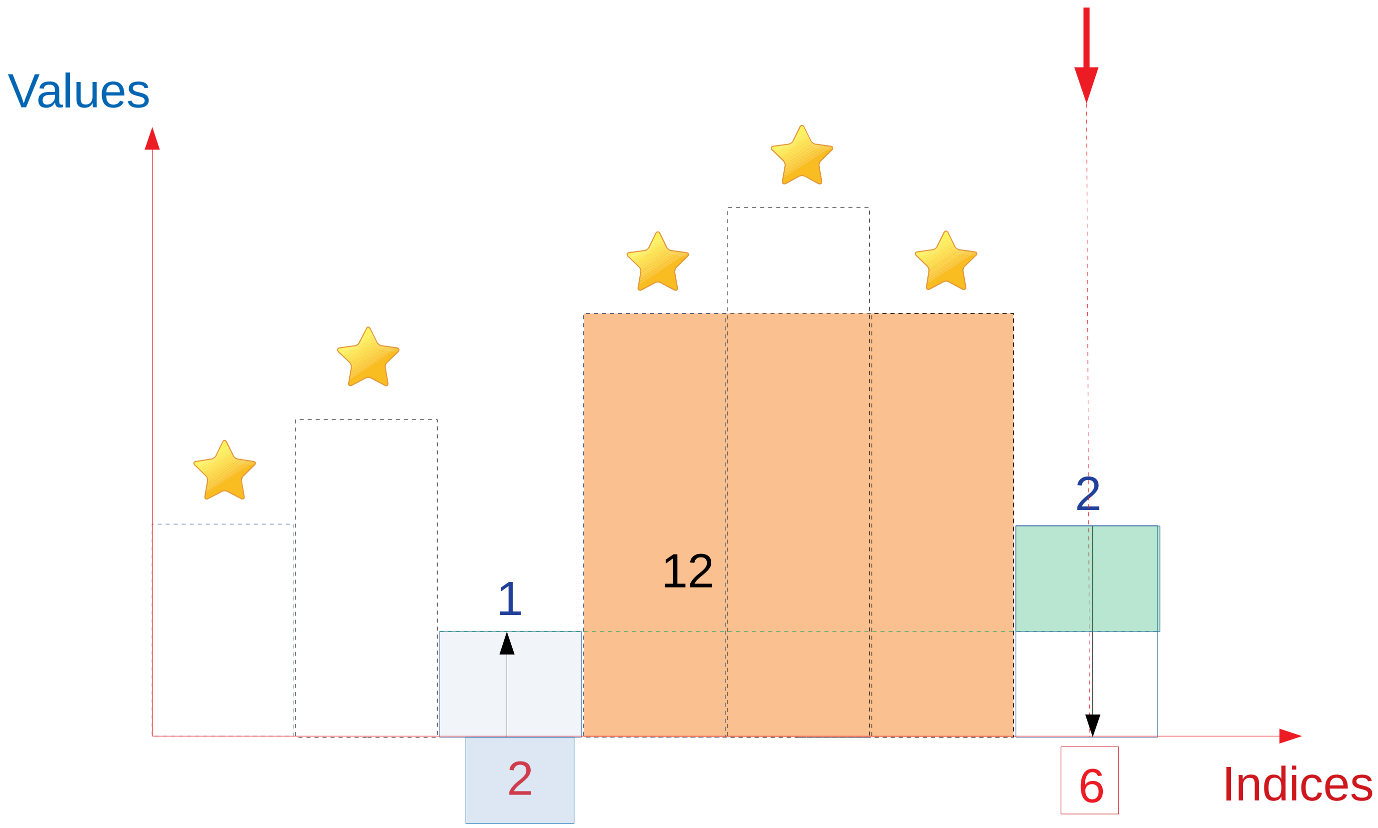
(6-1) – 2 = 3



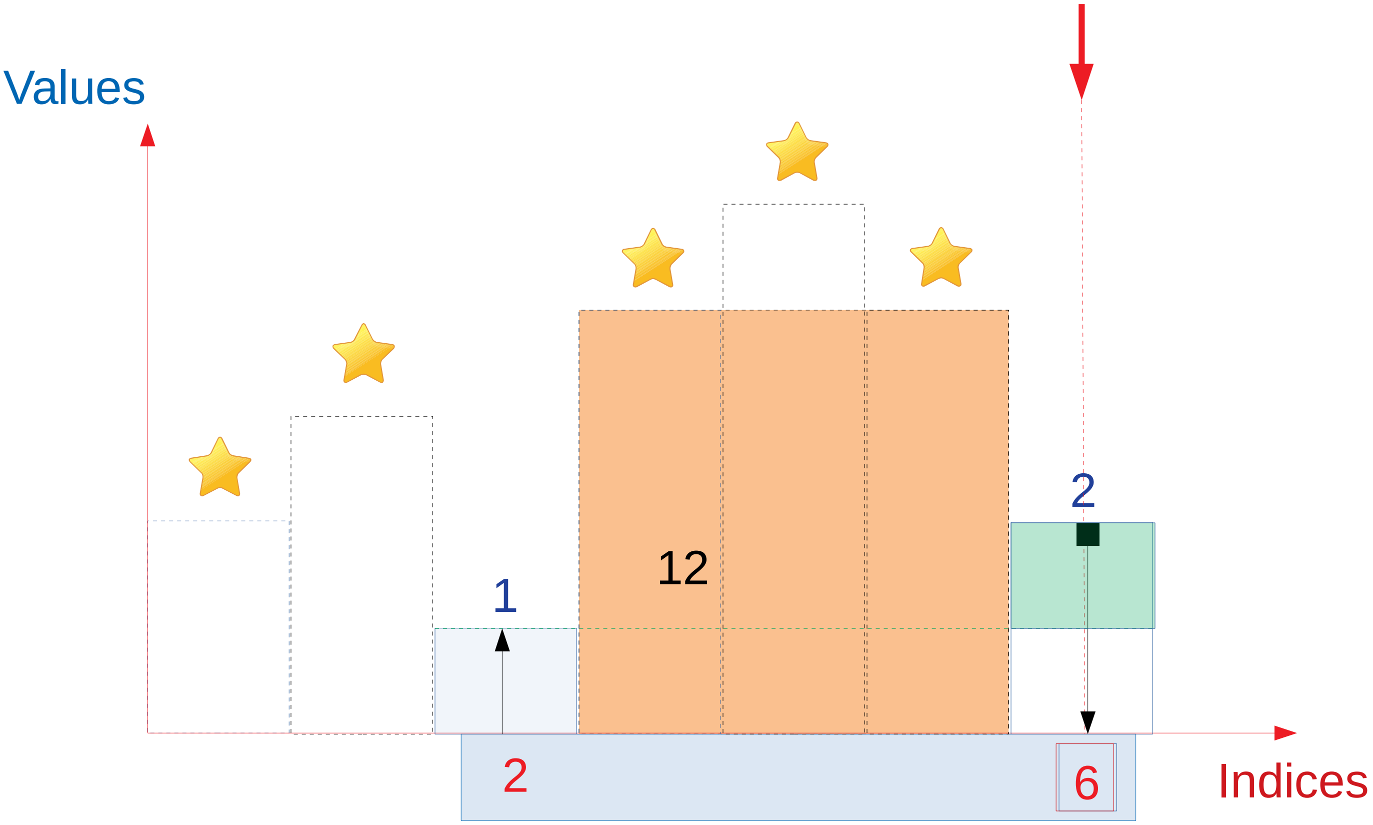
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



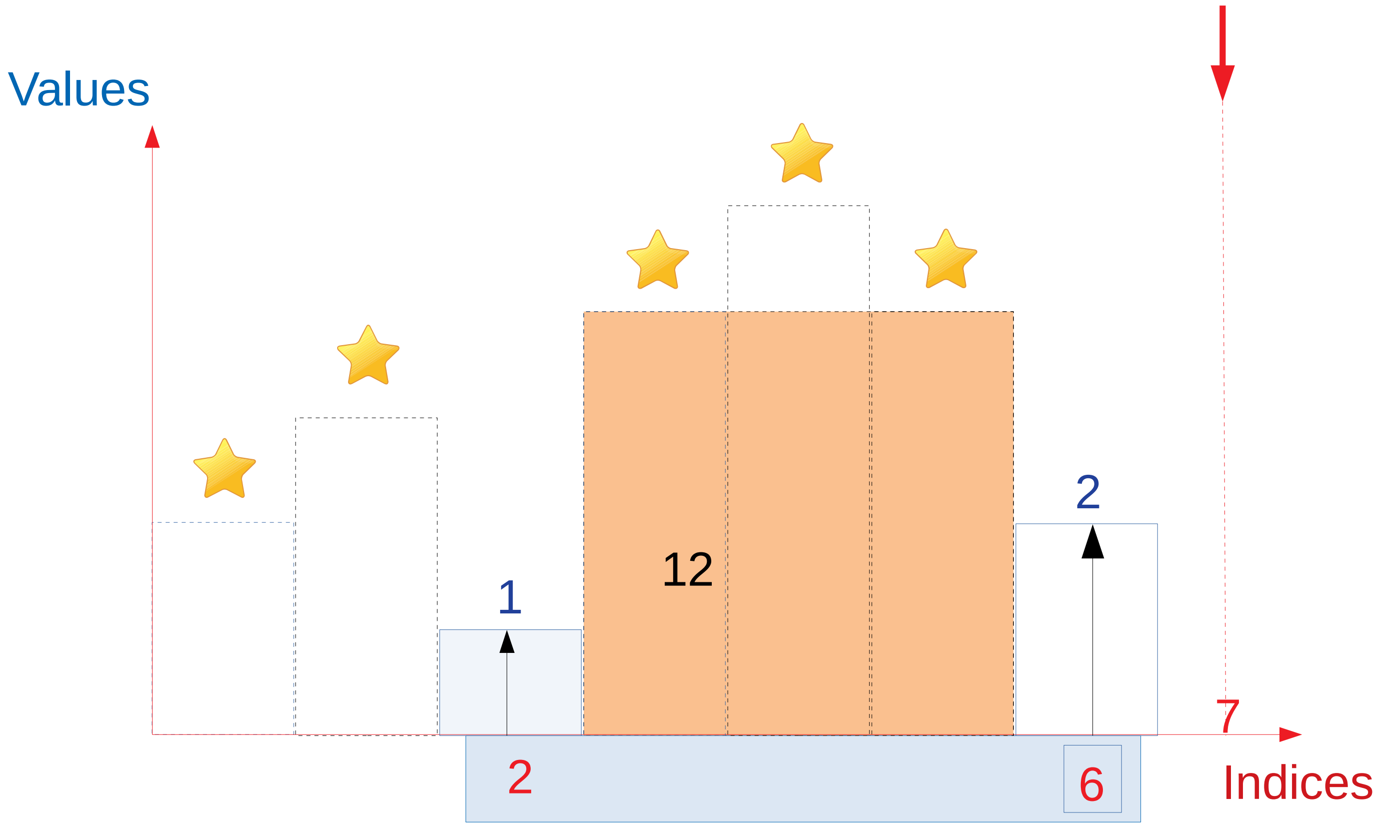
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



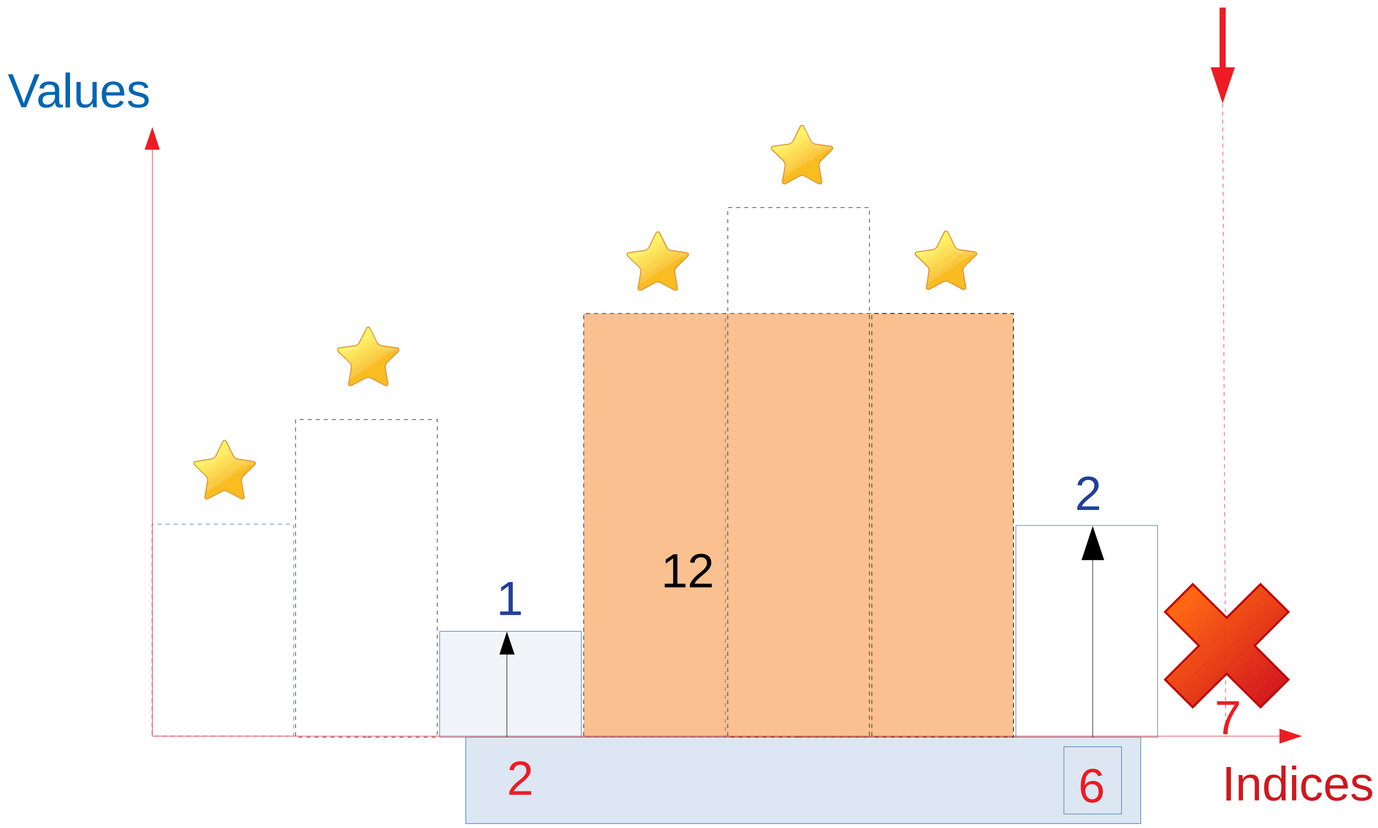
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
             (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



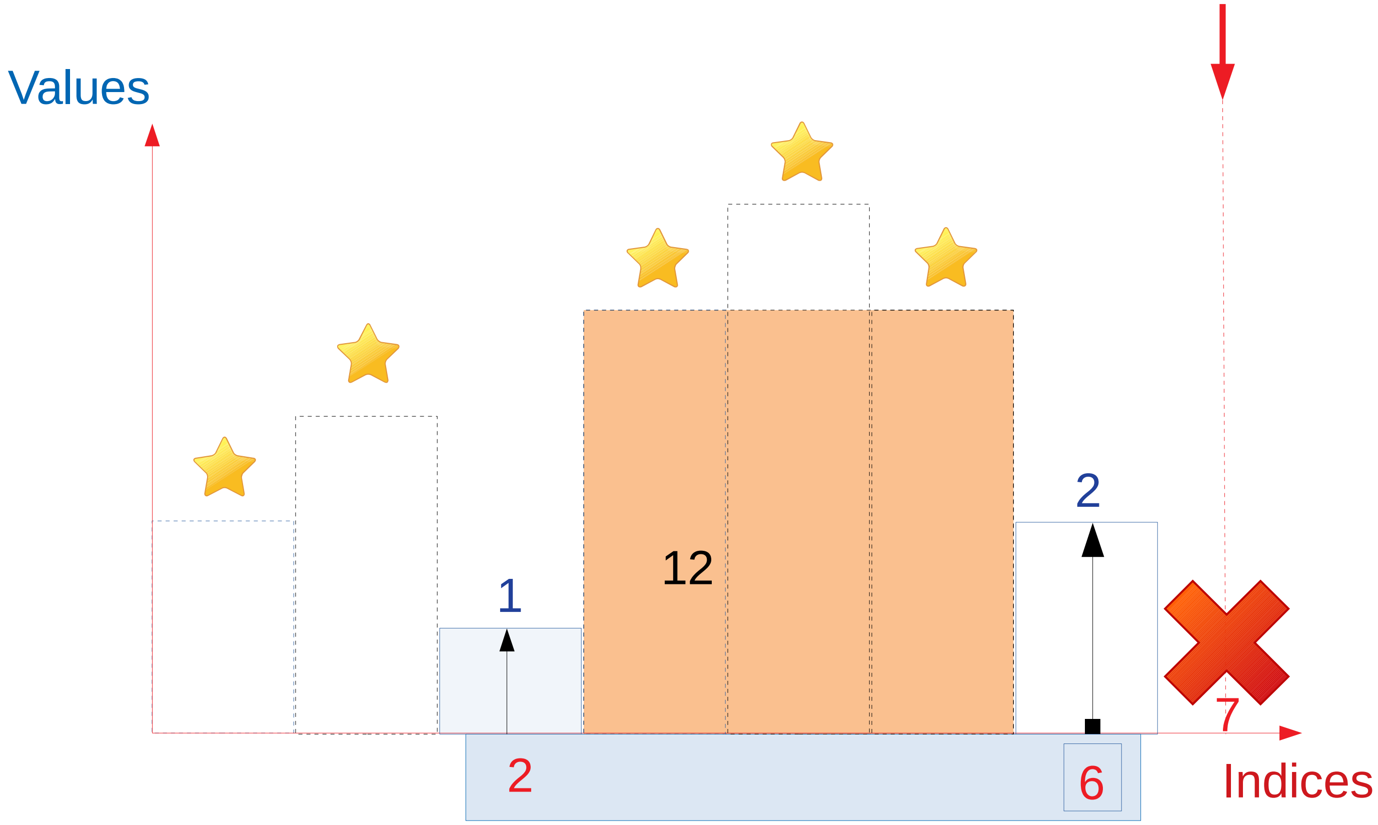
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
    (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



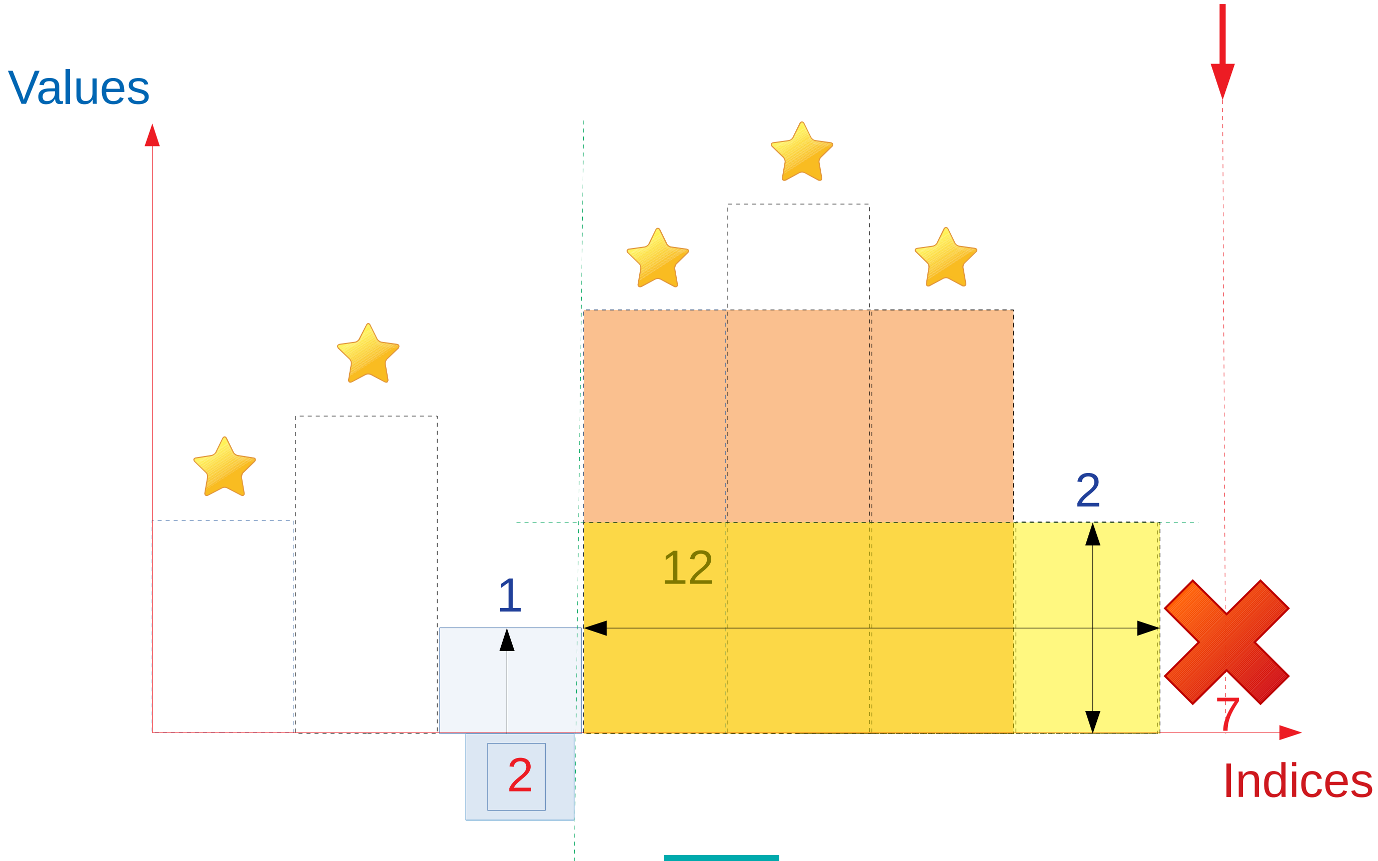
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



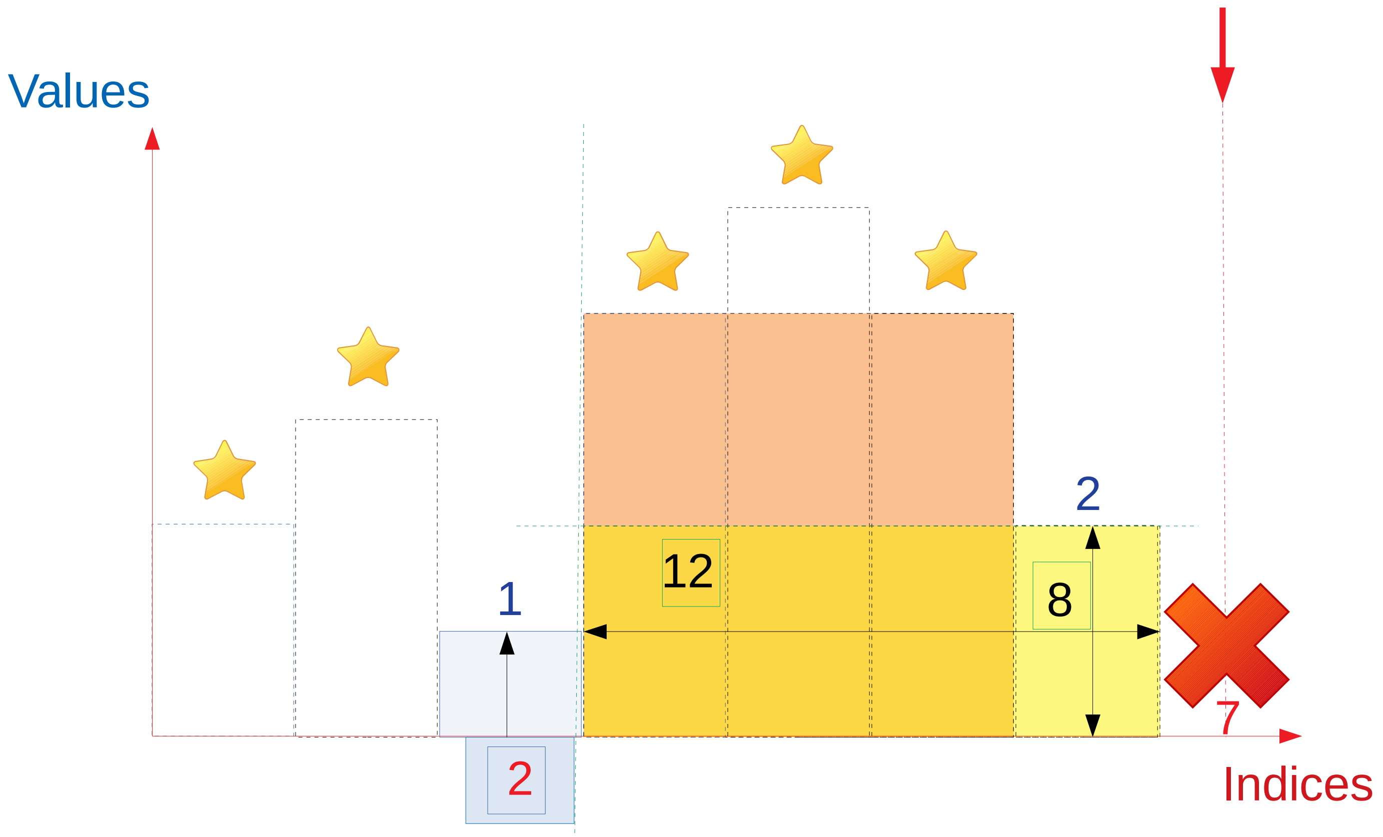
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
(isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                  else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```

2

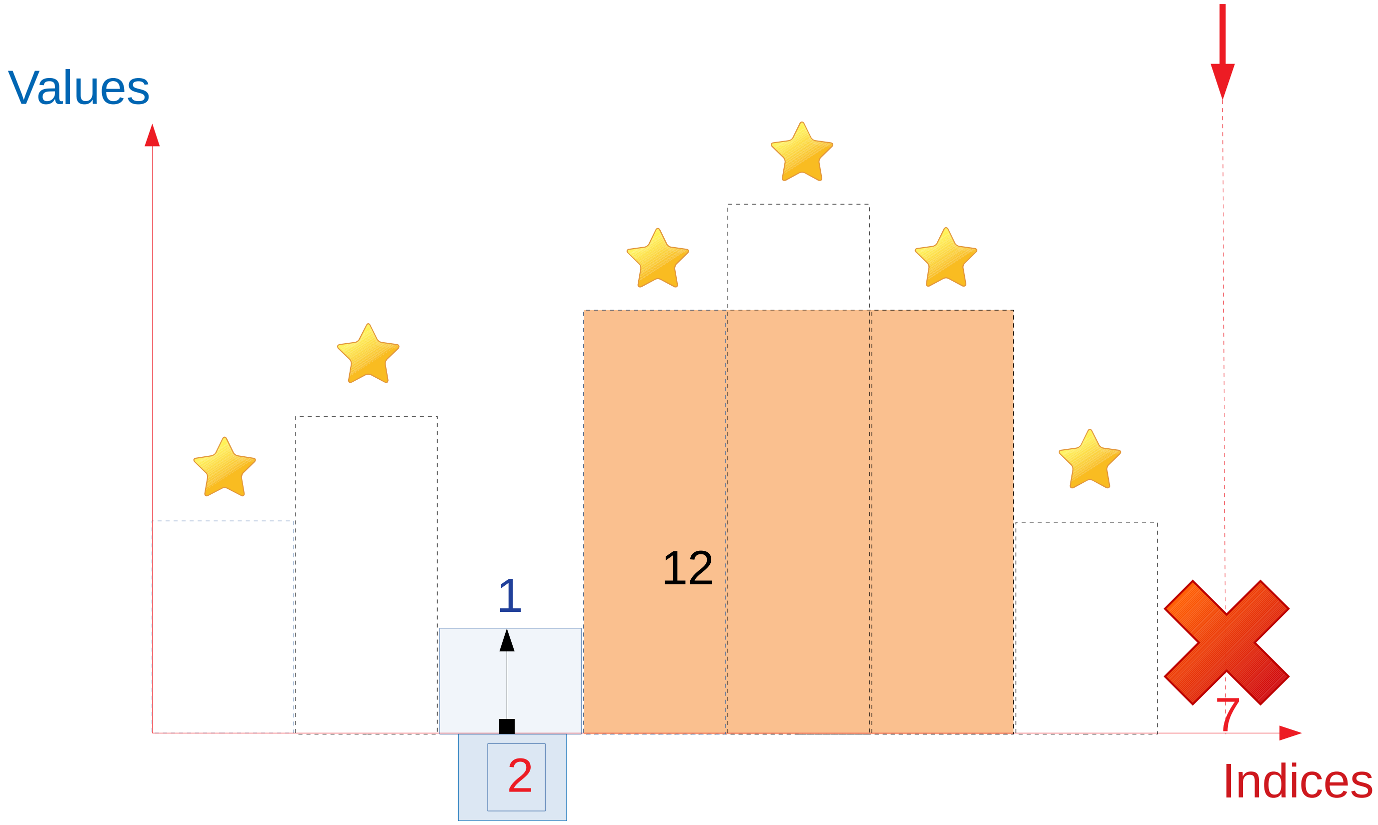
(7-1) – 2 = 4



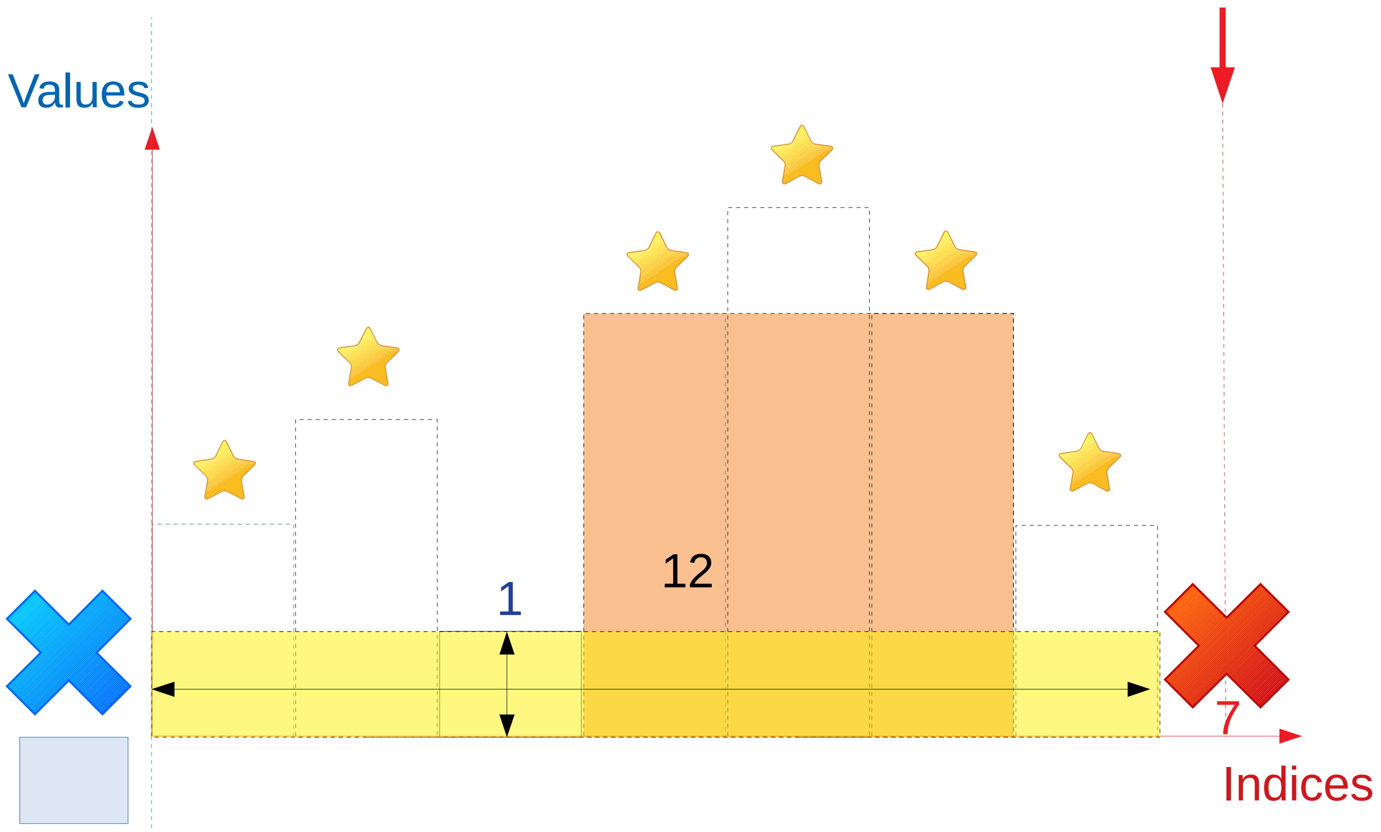
```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```

2

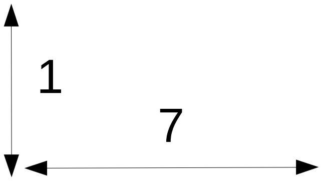
(7-1) – 2 = 4

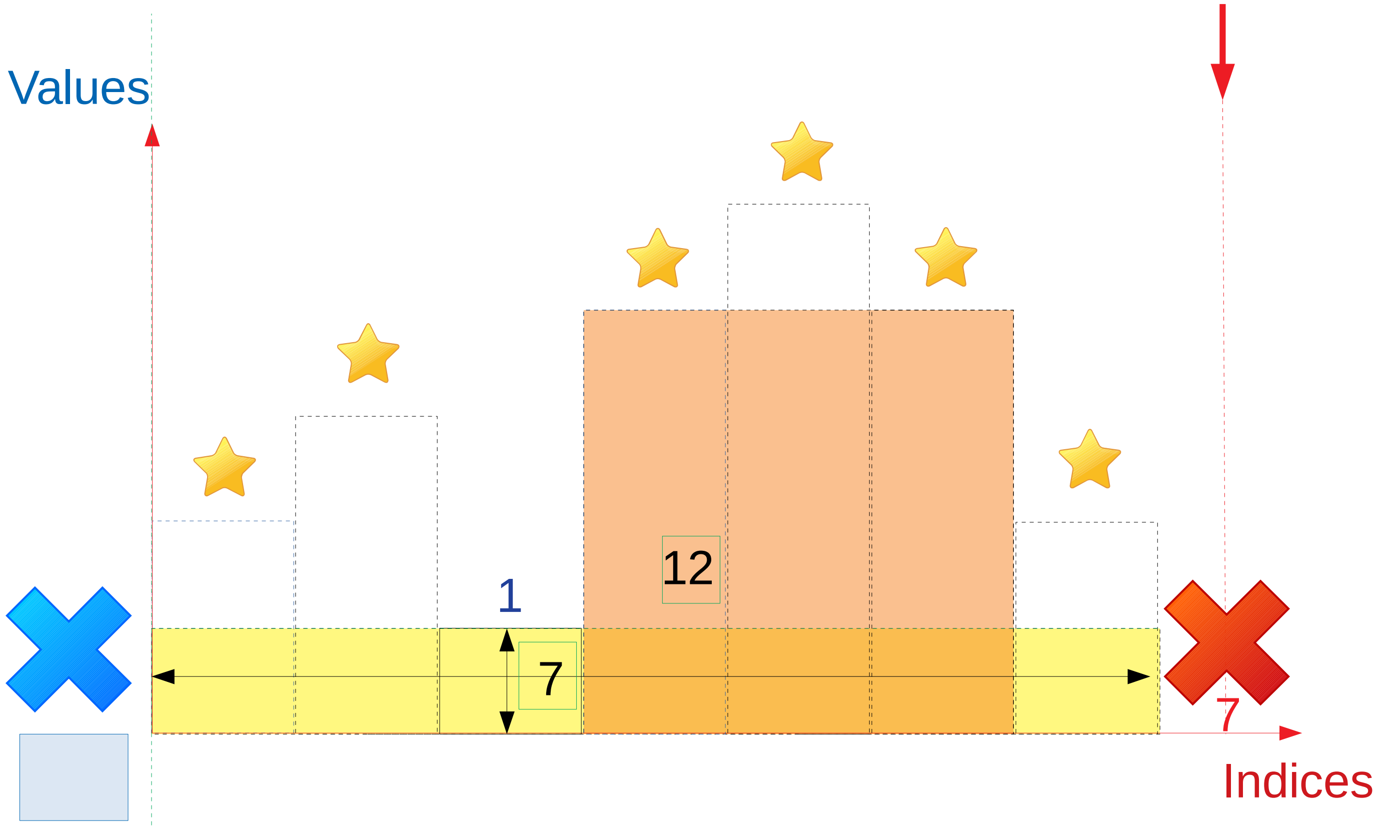


```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
      if (i < len_hist) and
   (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
      else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```

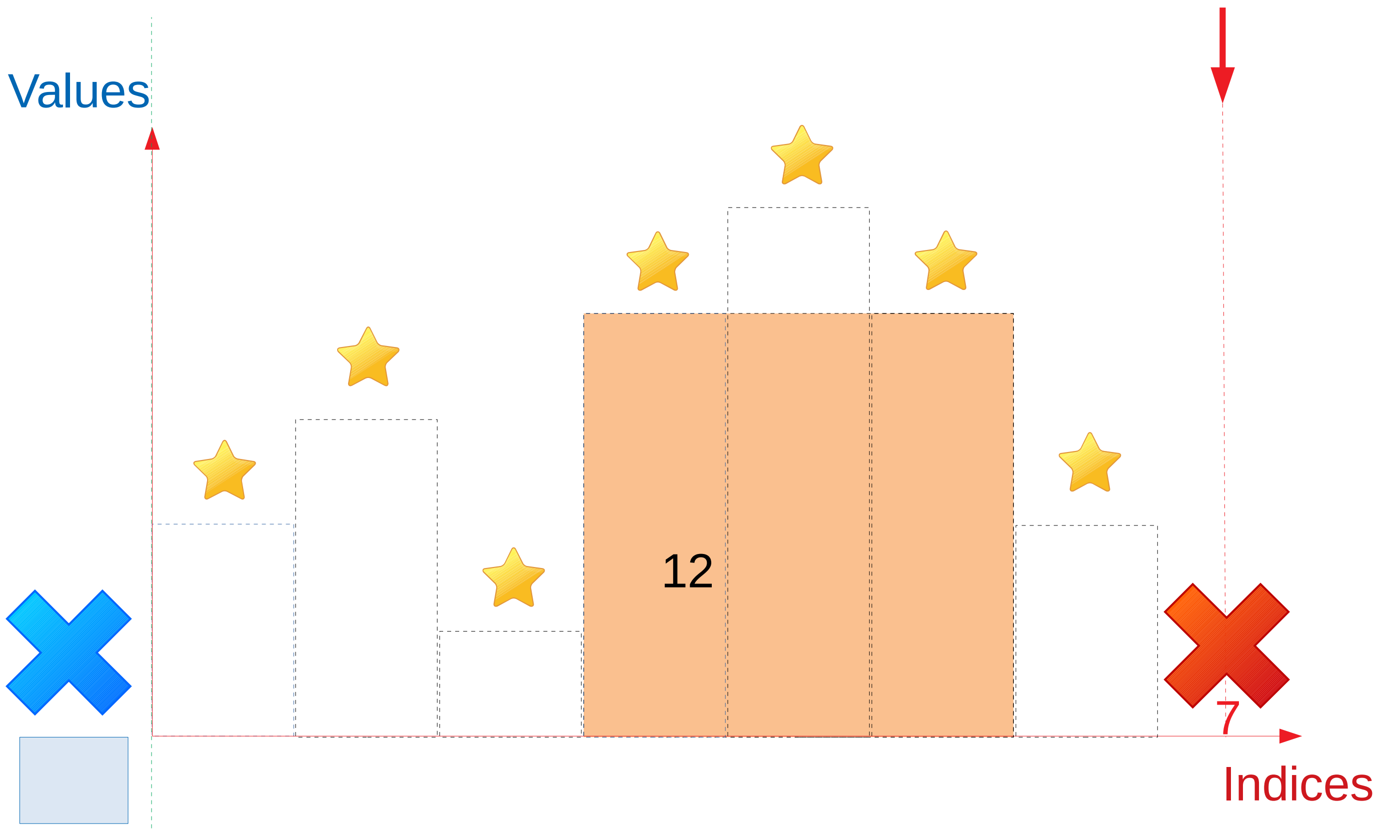


```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```

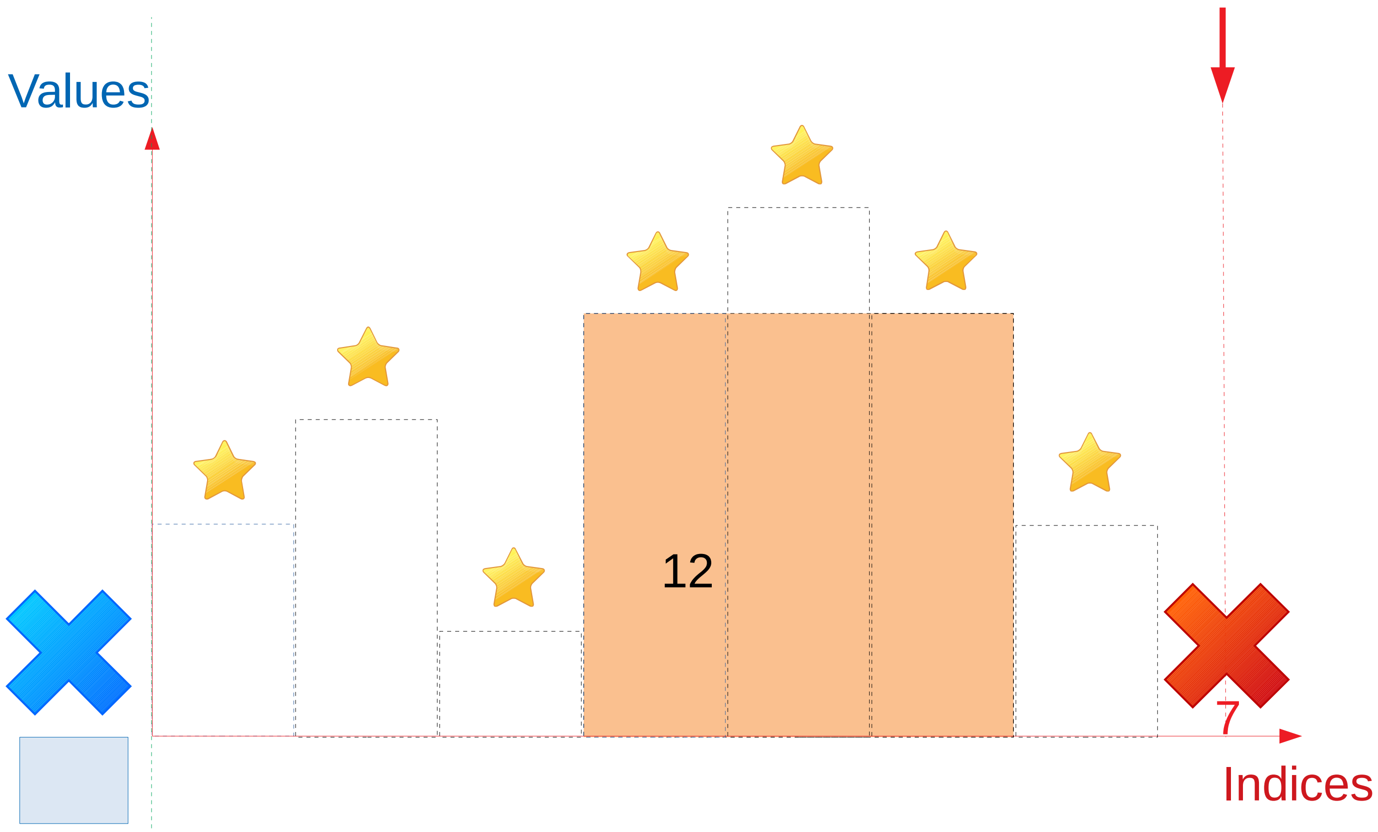




```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```



```
def getMaxAreaFromHisto(hist: List[Any]) -> int:
    len_hist = len(hist)
    s = []  # type: List[Any]
    max_area = 0
    i = 0
    while (i < len_hist) or isNotEmpty(s):
        if (i < len_hist) and
       (isEmpty(s) or hist[getTop(s)] <= hist[i]):
            s.append(i)
            i += 1
        else:
            tp = s.pop()
            area_with_top = hist[tp] *
              (i if isEmpty(s)
                 else (i - 1) - getTop(s))
            max_area = max(max_area, area_with_top)
    return max_area
```