Algorithm 1: tope del contenedor de pilas

```
Function tope(Pila)
    i → 0
    s → 0
    while i < NumPilas − 1 do
        s → Tope[i] − Base[i] + s
        i → i + 1
    end
return s
```

Algorithm 2: Obtener Menor indice j

```
Function ObtenerMenorj(entero i)
    j → 0
    if i < NumPilas − 2 ∧ i ≥ 0 then
        j = i + 1
        while j < NumPilas − 1 ∧ Base[j + 1] = Tope[j] ∨ j = NumPilas − 1 ∧ Tope[j] = Max − 1 do
            j → j + 1
        end
    end
return res(j, NumPilas)
```

Algorithm 3: Obtener Mayor indice j

```
Function ObtenerMayorj(entero i)
    j → −1
    if i ≥ 0 ∧ i < NumPilas then
        j = i
        while j > 0 ∧ Base[j] = Tope[j − 1] ∨ j ≥ 0 ∧ Base[i] = −1 do
            if (j > 0 ∧ Base[j] < 0) then
                j → −1
            end
        end
    else
        j → j − 1
    end
return j
```

Algorithm 4: Apilar

```
Procedure Apl(Pila, i, x)
    if i < NumPilas − 1 ∧ i ≥ 0 then
        if tope(Pila) < Max then
            if Tope[i] = Base[i + 1] then
                j → ObtenerMenorJ(i)
                if j > 0 then
                    for k = Base[i + 1] + 1 downto Tope[j] do
                        Pila[k + 1] = Pila[k]
                    end
                    for k = i + 1 to j do
                        Tope[k] = Tope[k] + 1
                        Base[k] = Base[k] + 1
                    end
                else
                    j = ObtenerMayorJ(i)
                    if j ≥ 0 then
                        for k = Base[j] + 1 to Tope[i] do
                            Pila[k − 1] = Pila[k]
                        end
                        for k = j downto i do
                            T[k] = T[k] − 1
                            B[k] = B[k] − 1
                        end
                    end
                end
            end
            Tope[i] = Tope[i] + 1
            Pila[T[i]] = x
        else
            print Llena
        end
    end
```

Algorithm 5: Apilar en la ficticia

```
Procedure Apl2(Pila,x)
    if tope(x) < Max then
        i → ObtenerMayorJ(13)
        for j = Base[i] + 1 to j ≤ Tope[13] do
            Pila[j − 1] = Pila[j]
        end
        for j = 13 downto i do
            Tope[j] = Tope[j] − 1
            Base[j] = Base[j] − 1
        end
        Tope[13] = Tope[13] + 1
        Pila[T[13]] = x
    end
```

Algorithm 6: Desapilar

```
Function DsApl(Pila,i)
    x → 0
    if i ≥ 0 ∧ i < NumeroPilas then
        if ¬PilaVacia(Pila, i) then
            x = Pila[Tope[i]]
            T[i] = T[i] − 1
        end
    end
return x
```