STOS (lista lifo: last in - first out, STACK)
Stos jest to lista, w której wstawianie oraz usuwanie elementów odbywa się tylko na jednym końcu zwanym wierzchołkiem stosu (TOP).
Z reguły identyfikujemy wierzchołek z pierwszym elementem listy.
Operacje na stosie:

- **MAKENULL(S)** – uczyń stos pustym.
- **TOP(S)** – zwróć element znajdujący się na wierzchołku stosu.
- **POP(S)** – Usuń element znajdujący się na wierzchołku stosu.
- **PUSH(x,S)** – umieść element x na wierzchołku stosu S.
  Element, który znajdował się poprzednio na wierzchołku stosu staje się wtedy następnym po x, itd.
- **EMPTY(S)** – zwraca true, jeśli S jest stosem pustym,
  w przeciwnym razie zwraca false

Zad.1. Wykorzystać operacje na klasie lista (reprezentacja wskaźnikowa) do zaimplementowania operacji na stosie.

Mamy klasę lista:

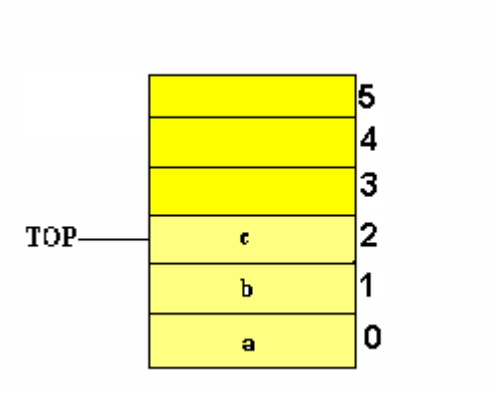
```
class Lista
{protected :
        position l;//wskaźnik do pierwszego elementu
public:
        void Insert(elementtype x, position p);
        void Delete(position p);
        elementtype Retrieve(position p);
        position Locate(elementtype x);
        position First();
        position Next(position p);
        position Previous(position p);
        position END();
        …};
```

Tworzymy obiekt tej klasy: List *S=new Lista;
Gdy mamy już obiekt S typu Lista, chcemy na nim wykonać żądane operacje.

Top:
Pop:
Push:
Empty, Makenull

Tablicowa implementacja stosu:

- Każda z omówionych implementacji listy może zostać użyta do realizacji stosu.
- Ale implementacja stosu za pomocą listy tablicowej jest zbyt kosztowna, gdyż każda operacja POP i PUSH wymaga przesuwania wszystkich elementów stosu.
- Dlatego lepsza jest następująca implementacja:
  -



  - Kolejne elementy stosu znajdują się w komórkach S[0], S[1], S[TOP]
  - Stos jest pusty, gdy TOP=-1

Zad.2. Zaimplementować metody klasy Stos (reprezentacja tablicowa)

```
const maxlegth=20;
typedef int elementtype;
typedef int position;

class Stos
{
        int S[maxlegth] ;
        position Top;//szczyt stosu
public:
        Stos();
        void Push(elementtype x);
        void Pop();
        bool Empty();
        elementtype TopElem();
        void Makenul();
};
```

Uwagi:

1. Konstruktor ustawia Top na -1
2. PUSH – sprawdzamy, czy jest miejsce w tablicy, jeśli jest, wstawiamy element na szczyt stosu, uaktualniając pole TOP
3. POP- sprawdzamy, czy stos nie jest pusty. Jeśli nie jest, usuwamy element ze szczytu stosu uaktualniając pole TOP.
4. Top_elem – sprawdzamy, czy stos nie jest pusty. Jeśli nie jest pusty, zwracamy wartość ze szczytu stosu