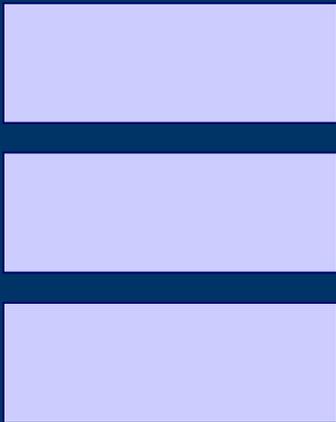


Pile



PILA (Stack)

- Una pila è un insieme ordinato, non limitato, di elementi dello stesso tipo nella quale il primo elemento memorizzato e' l'ultimo elemento ad essere recuperato (LIFO=Last Input First Output)



PILA (Stack)

■ Il tipo PILA è un ADT $\langle S, F, C \rangle$ dove

– $S = \{\text{pila}, \text{atomo}, \text{boolean}\}$

pila è il dominio di interesse atomo è il dominio degli elementi che formano le liste

$F = \{\text{push}, \text{top}, \text{empty}, \text{pop}\}$

$\text{push} : \text{atomo} \times \text{pila} \rightarrow \text{pila}$ inserisce un elemento in cima alla pila

$\text{top} : \text{pila} \rightarrow \text{atomo}$ ritorna l'elemento in cima alla pila

$\text{empty} : \text{pila} \rightarrow \text{boolean}$ ritorna il valore vero se la pila è vuota

$\text{pop} : \text{pila} \rightarrow \text{pila}$ ritorna la pila privata dell'elemento in cima

$C = \text{pila vuota}$, è la costante che denota la coda priva di elementi

Implementazione concatenata di pile

```
typedef int TAtomo;
typedef struct Stelem {
    struct Selem *next;
    TAtomo info;
} elemPila, *TPila;
```

```
int pop (TPila *PP){
    TPila aux;
    if (null(*PP)) return 0;
    aux = *PP;
    *PP = *PP->next;
    free(aux);
    return 1;
}
```

Implementazione concatenata di pile

```
int null( TPila P ) { return P == NULL;}
TAtomo top( Pila P ) {
    if( null(P) ) return NULL;
    return P -> info;
}
int push( TPila *PP, TAtomo A ) {
    Tpila *aux;
    aux = (TPila)malloc(sizeof(elpila));
    if (!aux) return 0;
    aux ->info = A;
    aux -> next = *PP;
    *PP = aux;
    return 1;
}
```
