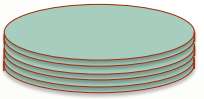
ADT *STACK (PILA)*

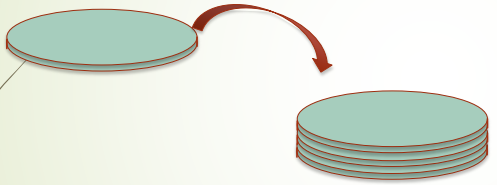
ADT *Stack (pila)*

- Una ***pila*** è una sequenza di elementi di un determinato tipo, in cui è possibile aggiungere o togliere elementi esclusivamente da un unico lato (***top*** dello stack).
  - La pila è una struttura dati *lineare* a *dimensione variabile* in cui si può accedere direttamente solo al ***primo*** elemento della lista.
  - Non è possibile accedere ad un elemento diverso dal primo **se non dopo** aver eliminato tutti gli elementi che lo precedono (inseriti dopo).
  - Lista gestita con la modalità ***LIFO (Last-in-first-out)*** cioè l'ultimo elemento inserito nella sequenza sarà il primo ad essere eliminato.

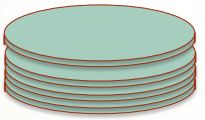
ADT stack
Esempio: una pila di monete



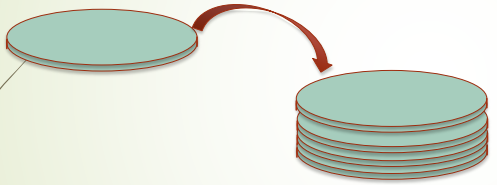
ADT stack
Esempio: una pila di monete



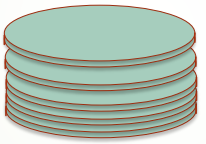
ADT stack
Esempio: una pila di monete



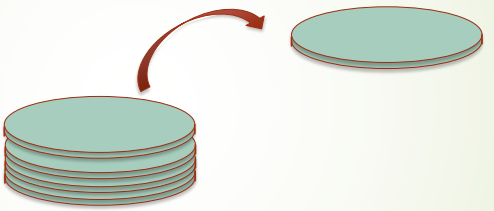
ADT stack
Esempio: una pila di monete



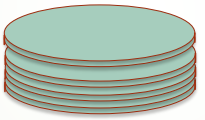
ADT stack
Esempio: una pila di monete



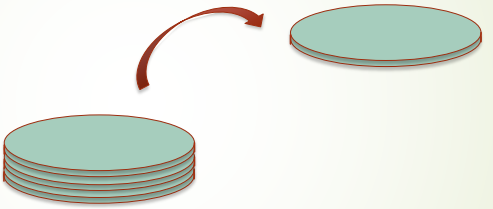
ADT stack
Esempio: una pila di monete



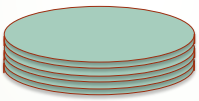
ADT stack
Esempio: una pila di monete



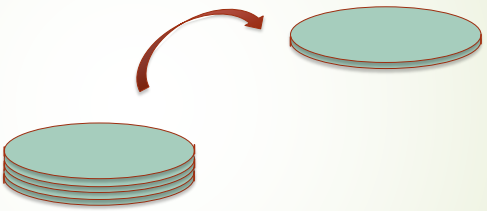
ADT stack
Esempio: una pila di monete



ADT stack
Esempio: una pila di monete



ADT stack
Esempio: una pila di monete



ADT stack

Esempio: una pila di monete

ADT: Stack

Sintattica	Semantica
Nome del tipo: Stack Tipi usati: Item, boolean	Dominio: insieme di sequenze $S=a_1,\ldots,a_n$ di tipo Item L'elemento **nil** rappresenta la pila vuota
newStack() → Stack	newStack() → s • Post: s = nil
isEmptyStack(Stack) → boolean	isEmpty(s) → b • Post: se s=nil allora b = true altrimenti b = false
push(Stack, Item) → Stack	push(s, e) → s' • Post: s = <a1, a2, … an> AND s' = <e, a1, …, an>
pop(Stack) → Stack	pop(s) → s' • Pre: s = <a1, a2, …, an> n>0 • Post: s' = <a2, …, an>
top(Stack) → Item	top(s) → e • Pre: s = <a1, a2, …, an> n>0 • Post: e = a1

ADT *Stack*
Implementazione

- Tra le possibili implementazioni, le più usate sono realizzate tramite:
  - **Array**
  - **Lista concatenata**

ADT *Stack*
Implementazione con Array

- Lo stack è implementato come un puntatore ad una struct stack che contiene due elementi:
  - Un array di MAXSTACK elementi
  - Un intero che indica la posizione del top dello stack
  - Quando lo stack si riempie, non è possibile eseguire l'operazione push …

ADT *Stack*
Implementazione con Lista

- Il tipo stack è definito come un puntatore ad una struct che contiene
  - Un elemento items di tipo list
  - Non serve più nemmeno l'intero MAXSTACK che indica la capienza massima dello stack …
  - Anche se abbiamo un solo elemento nella struct, continuiamo a definire il tipo stack come puntatore a struct stack per non cambiare la definizione nell'header file