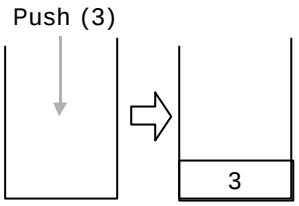
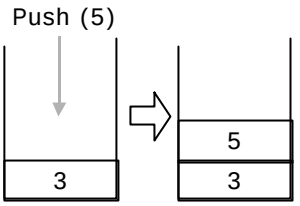
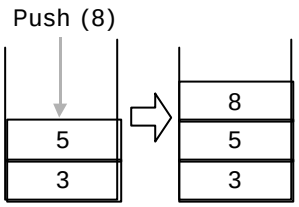
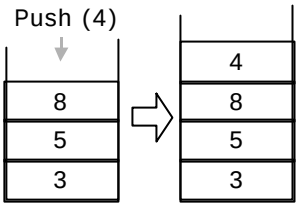
STACK (PILE)

Uno **stack** o **pila** e' una struttura che memorizza dati omogenei, permettendone l'inserimento e l'estrazione di essi in modalita' **LIFO** (last-in-first-out)

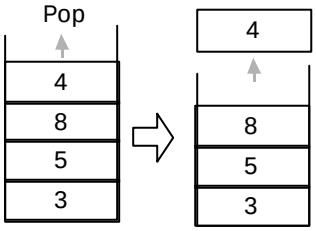


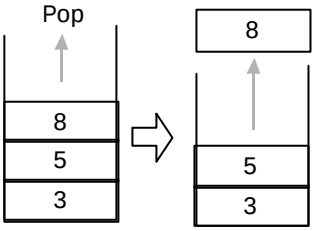


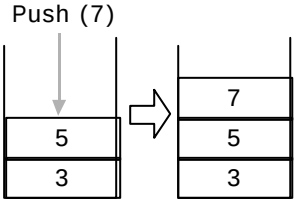


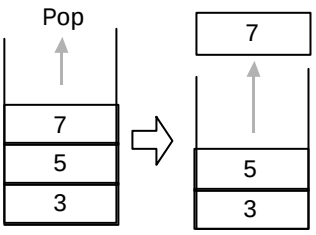


STACK (PILE)









STACK (PILE)

Su una pila vengono definite due funzioni:

PUSH (elemento) inserisce l'elemento in testa allo stack (se questo non e' pieno)

POP () → *elemento* Estrae l'elemento in testa allo stack (se questo non e' vuoto), e lo restituisce

IMPLEMENTAZIONE DI UNO STACK

Usiamo un vettore statico e definiamo la seguente struttura dati:

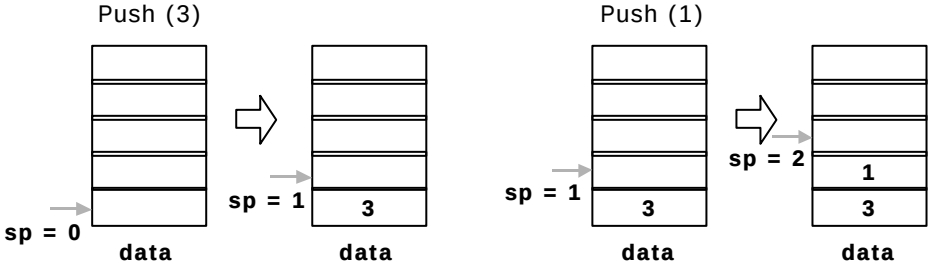
```
#define STACK_SIZE ...

typedef struct {
  int sp;
  int data [STACK_SIZE];
} t_stack;
```

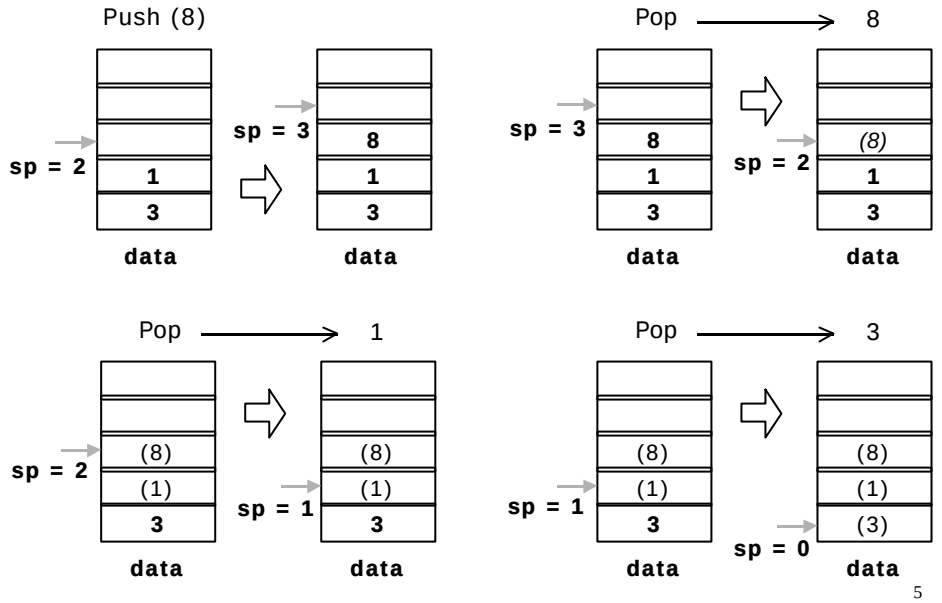
Dimensione massima dello stack

Indice del prossimo elemento dove fare l'inserzione (stack pointer)

Array che contiene i dati dello stack



IMPLEMENTAZIONE DI UNO STACK



IMPLEMENTAZIONE DI UNO STACK

```
typedef struct {
  int   sp;
  int   data [STACK_SIZE];
} t_stack;

void push (t_stack * s, int value)
{
  s->data [s->sp] = value;
  s->sp ++;
}

int pop (t_stack * s)
{
  s->sp --;
  return s->data [s->sp];
}
```

IMPLEMENTAZIONE DI UNO STACK

```
typedef struct {
  int   sp;
  int   data [STACK_SIZE];
} t_stack;

t_stack init_stack (void)
{
  t_stack new_stack;
  new_stack.sp = 0;
  return new_stack;
}

int is_stack_empty (t_stack s)
{
  return s.sp == 0;
}

int is_stack_full (t_stack s)
{
  return s.sp == STACK_SIZE;
}
```


IMPLEMENTAZIONE DI UNO STACK

```
typedef struct {
  int   sp;
  int   data [STACK_SIZE];
} t_stack;

void dump_stack (t_stack s)
{
  int i;
  for (i = s.sp - 1; i >= 0; i--)
    printf ("%d ", s.data [i]);
  printf ("\n");
}
```

IMPLEMENTAZIONE DI UNO STACK

```
int main (int argc, char * argv[])
{
  t_stack mystack;
  int choice;

  mystack = init_stack ();
  do {
    printf ("0. Exit\n");
    printf ("1. Push elemento\n");
    printf ("2. Pop elemento\n");
    printf ("3. Stampa stack\n");
    scanf ("%d", &choice);
...
```


IMPLEMENTAZIONE DI UNO STACK

```
...
    switch (choice) {
      case 1:
        {
          int elem;
          printf ("Inserisci l'elemento :");
          scanf ("%d", &elem);
          if (is_stack_full (mystack) == 0)
            push (&mystack, elem);
          else
            printf ("Stack pieno, inserimento fallito\n");
        }
        break;
...
```

IMPLEMENTAZIONE DI UNO STACK

```
...
      case 2:
        {
          int elem;
          if (is_stack_empty (mystack) == 0) {
            elem = pop (&mystack);
            printf ("Pop riuscito = %d\n", elem);
          }
          else
            printf ("Stack vuoto, pop fallito\n");
        }
        break;
      case 3:
        dump_stack (mystack);
        break;
    }
  } while (choice != 0);
}
```


A CHE SERVE UNO STACK?