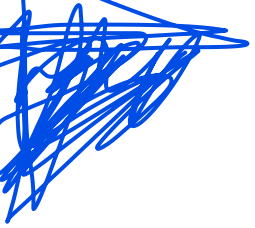


LABORATORIO DI ALGORITMI E STRUTTURE DATI

TURN0 2 - CORSO A

OBIETTIVI

- Realizzazione e sperimentazione **degli algoritmi** e delle **strutture dati**
- In «laboratorio»
  - Presentazione di alcuni **esempi**
  - Presentazione degli **esercizi**
  - Lavoro in gruppo!
  - Supporto del docente

PROGETTO DI LABORATORIO

- **4 esercizi da svolgere in gruppo (1-3 persone)**
  - Tutte appartenenti allo stesso turno
  - Componenti del gruppo entro il terzo incontro (**PERMESSI NEL GIT**)
  - Cambiamenti non saranno ammessi
- https://gitlab2.educ.di.unito.it/drago/laboratorio-algoritmi-2021-2022
- Git per gestione/condivisione/versioning del codice
- **Corretto utilizzo di Git è valutato in sede d'esame**

ESAME

- **Relazione** sulla sperimentazione associata agli esercizi **1, 2 e 4**, con discussione delle scelte operate in fase di implementazione
- Esame orale con discussione del progetto di laboratorio
  - Valutazione su 30+1 punti
  - Necessaria la sufficienza sia prova scritta che laboratorio
  - Voto finale per il corso da 9 CFU = media pesata del voto della prova scritta e del voto di laboratorio

ESAME

- **Regole sulla pagina Git e sulla pagina Moodle del corso**
- Iscrizione in 2 passi:
  - Tramite myUnito all'appello selezionato
  - Tramite risorsa su i-learn per la prenotazione della discussione
- I membri di uno stesso gruppo devono appartenere tutti allo stesso turno di laboratorio
- **Consegna relazione entro e non oltre la data della prova scritta**
- Non si può sostenere la prova scritta senza la consegna del progetto di laboratorio
- La prova orale va sostenuta nella medesima sessione della prova scritta superata
- **Validità del progetto di laboratorio: sessione gennaio-febbraio di questo anno accademico**

PROGETTO DI LABORATORIO

- Richiesto codice di «**buona qualità**»
  - Modulare
  - Ben commentato
  - Ben testato
- Implementazione degli **unit-test** degli algoritmi

PROGETTO DI LABORATORIO - Suggerimenti

«verificare che il codice sia suddiviso **correttamente**

in **package** o **moduli**»

PROGETTO DI LABORATORIO - Suggerimenti

```java
package orderedarray;

import java.util.ArrayList;
import java.util.Comparator;

/**
 *
 * @author pozzato
 * @param <T>: type of the ordered array elements
 */
public class OrderedArray<T> {
    private ArrayList<T> array = null;
    private Comparator<? super T> comparator;

    /**
     * It creates an empty ordered array and returns the created array.
     * It accepts as input a comparator implementing the precedence raltion between the array elements.
     * @param comparator: a comparator implementing the precedence relation between the array elements
     */
    public OrderedArray(Comparator<? super T> comparator){
        this.array = new ArrayList<>();
        this.comparator = comparator;
    } // OrderedArray
```

PROGETTO DI LABORATORIO - Suggerimenti

```
1   package orderedarray;
2
3   import java.util.ArrayList;
4   import java.util.Comparator;
5
6   /**
7    *
8    * @author pozzato
9    * @param <T>: type of the ordered array elements
10   */
11  public class OrderedArray<T> {
78  } // class
79
80
81  public class OrderedArrayUsageJava {
82
83      private static final Charset ENCODING = StandardCharsets.UTF_8;
84
85      private static void printArray(OrderedArray<Record> orderedArray) throws OrderedAr Exception{
86          Record currRec = null;
87          int sizeArr;
88
89          System.out.println("\nORDERED ARRAY OF RECORDS\n");
90          sizeArr = orderedArray.size();
```

PROGETTO DI LABORATORIO - Suggerimenti

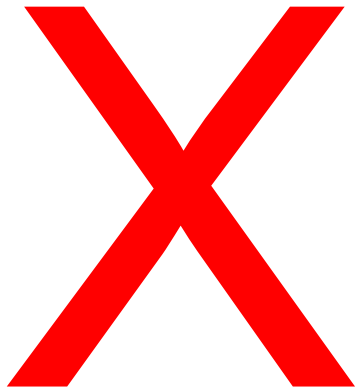
«aggiungere un **commento**, prima di una definizione, che spiega il **funzionamento dell'oggetto definito**. Evitare quando possibile di commentare direttamente il codice in sé (se il codice è ben scritto, i commenti in genere non servono)»

PROGETTO DI LABORATORIO - Suggerimenti

```
62      // It returns the position where the new element must be inserted
63      private int getIndexInsert(T element){
64          int index = 0;
65          boolean cont = true;
66          T currEl = null;
67          while (cont && (index < (this.array).size())){
68              currEl = (this.array).get(index);
69              if ((this.comparator).compare(element,currEl)<0)
70                  cont = false;
71              else
72                  index++;
73          }  // while
74          return index;
75      } // getIndexInsert
```

PROGETTO DI LABORATORIO - Suggerimenti

```
62      // It returns the position where the new element must be inserted
63      private int getIndexInsert(T element){
64          int index = 0;
65          boolean cont = true;
66          T currEl = null;
67          // repeat while cont is true and index is inside array range
68          while (cont && (index < (this.array).size())){
69              currEl = (this.array).get(index);
70              if ((this.comparator).compare(element,currEl)<0)
71                  // faccio cont uguale a falso
72                  cont = false;
73              else
74                  // incremento l'indice
75                  index++;
76          }  // while
77          return index;
78      } // getIndexInsert
```



PROGETTO DI LABORATORIO - Suggerimenti

«**la lunghezza di un metodo/funzione** è un campanello di allarme: se essa cresce troppo, probabilmente è necessario rifattorizzare il codice spezzando la funzione in più parti.

https://en.wikipedia.org/wiki/Spaghetti_code

Spaghetti code is a pejorative phrase for unstructured and difficult-to-maintain source code. Spaghetti code can be caused by several factors, such as volatile project requirements, lack of programming style rules, and software engineers with insufficient ability or experience.[1]

PROGETTO DI LABORATORIO - Suggerimenti

«sono accettabili **commenti** in italiano,

sebben siano preferibili in **inglese**»

«tutti i nomi

(e.g., **nomi di variabili, di metodi, di classi**, etc.)

devono essere **significativi** e in **inglese**»

PROGETTO DI LABORATORIO - Suggerimenti

```
5  struct _OrderedArray {
6    void** array;
7    size_t size;
8    size_t capacity;
9    OrderedArrayCompareType compare;
10 };
11
12 OrderedArray* OrderedArray_new(size_t capacity, OrderedArrayCompareType compare) {
13   OrderedArray* result = (OrderedArray*) malloc(sizeof(OrderedArray));
14   result->array = (void**) malloc(sizeof(void*)*capacity);
15   result->size = 0;
16   result->capacity = capacity;
17   result->compare = compare;
18   return result;
19 }
20
```

PROGETTO DI LABORATORIO - Suggerimenti

```
5  struct OA {
6    void** a;
7    size_t s;
8    size_t c;
9    OACT cp;
10 };
11
12 OA* OA_new(size_t c, OACT cp) {
13   OA* r = (OA*) malloc(sizeof(OA));
14   r->a = (void**) malloc(sizeof(void*)*c);
15   r->s = 0;
16   r->c = c;
17   r->cp = cp;
18   return r;
19 }
```

PROGETTO DI LABORATORIO - Suggerimenti

«il codice deve essere correttamente **indentato**;

impostare l'indentazione a 2 o 4 caratteri e

impostare l'editor in modo che inserisca ‘**soft tabs**’

(i.e., deve inserire il numero corretto di spazi invece

che un carattere di tabulazione)»

PROGETTO DI LABORATORIO - Suggerimenti

```
39 void* OrderedArray_at(OrderedArray* array, size_t position) {
40   if(position >= array->size ) {
41       fprintf(stderr, "Array index (%ld) out of bounds (0:%ld)\n", position, array->size);
42       exit(EXIT_FAILURE);
43   }
44   return array->array[position];
45 }
46
47 void OrderedArray_check_and_realloc(OrderedArray* array) {
48   if( array->capacity > array->size )
49 return;
50       array->capacity *= 2;
51     array->array = realloc(array->array, sizeof(void*) * array->capacity);}
52
53 void OrderedArray_insert(OrderedArray* array, void* element) {
54   OrderedArray_check_and_realloc(array);
55   size_t i;
56
57   for (i = OrderedArray_size(array); i > 0 && array->compare(array->array[i-1], element) > 0; --i
   ) {array->array[i] = array->array[i-1]; } array->array[i] = element; array->size += 1; return;
58 }
```

PROGETTO DI LABORATORIO - Suggerimenti

« Java: i nomi dei package sono tutti in minuscolo senza separazione fra le parole; i nomi dei tipi (classi, interfacce, ecc.) iniziano con una lettera maiuscola e proseguono in camel case (es. **TheClass**), i nomi dei metodi e delle variabili iniziano con una lettera minuscola e proseguono in camel case (es. **theMethod**), i nomi delle costanti

sono tutti in maiuscolo e in formato snake case (es. **THE_CONSTANT**)

PROGETTO DI LABORATORIO - Suggerimenti

```java
package orderedarray;

import java.util.ArrayList;
import java.util.Comparator;

/**
 *
 * @author pozzato
 * @param <T>: type of the ordered array elements
 */
public class OrderedArray<T> {
    private ArrayList<T> array = null;
    private Comparator<? super T> comparator;

    /**
     * It creates an empty ordered array and returns the created array.
     * It accepts as input a comparator implementing the precedence raltion between the array elements.
     * @param comparator: a comparator implementing the precedence relation between the array elements
     */
    public OrderedArray(Comparator<? super T> comparator){
        this.array = new ArrayList<>();
        this.comparator = comparator;
    } // OrderedArray
```

PROGETTO DI LABORATORIO - Suggerimenti

«C (ref. convenzioni progetto GTK+): macro e costanti sono tutti in maiuscolo e in formato snake case (es. **THE_MACRO**, **THE_CONSTANT**); i nomi di tipo (e.g. struct, typedefs, enums,...) iniziano con una lettera maiuscola e proseguono in camel case (e.g., **TheType**, **TheStruct**); i nomi di funzione iniziano con una lettera minuscola e proseguono in snake case (e.g., **the_function**()»

PROGETTO DI LABORATORIO - Suggerimenti

```
39 void* OrderedArray_at(OrderedArray* array, size_t position) {
40   if(position >= array->size ) {
41     fprintf(stderr, "Array index (%ld) out of bounds (0:%ld)\n", position, array->size);
42     exit(EXIT_FAILURE);
43   }
44   return array->array[position];
45 }
46
47 void OrderedArray_check_and_realloc(OrderedArray* array) {
48   if( array->capacity > array->size )
49     return;
50   array->capacity *= 2;
51   array->array = realloc(array->array, sizeof(void*) * array->capacity);
52 }
```