**Leonardo Motta – Matricola: S4952213 23/05/2023**

**Esercitazione BONUS – II parte**

**FORM – Analisi**

**Link repository assegnato:** <https://github.com/alessiofracchiolla/LabInfTel.git>

**1 Documentazione / Commenti**

**1.a** Generare la documentazione usando Doxygen e verificare se la documentazione prodotta permette di orientarsi nella struttura del progetto software. Indicare i punti che potrebbero essere migliorati.

**Risposta**:

La documentazione *Doxygen* viene generata correttamente e permette di orientarsi bene nella struttura del progetto. Il prompt del “.h” e del “.cpp” delinea bene le sezioni del codice, relative alle tipologie diverse di funzioni di accesso delle classi.

**1.b** Utilizzando la documentazione e i commenti inseriti nei file .ccp e .h verificare se il progetto software aderisce alle specifiche assegnate durante la prima settimana. Indicare le eventuali specifiche non rispettate.

**Risposta**:

* I commenti di ogni sezione del codice messi in evidenza sono puntuali e non ambigui.
* Nell’intestazione di ogni file “.h” e “.cpp” non è presente la specifica “*author*”.
* Il “main.cpp” è privo di documentazione *Doxygen*: in particolare in “main.cpp” sono presenti pochi commenti del flusso algoritmico seguito.

**1.c** Verificare che il file README introduca correttamente lo scopo del progetto software e che dia sufficienti informazioni per un corretto uso dell’interfaccia a riga di comando. Indicare eventuali mancanze e/o possibili migliorie.

**Risposta**:

File “README.md” non è presente nel *repository*.

**2 Compilazione e prima sessione di test**

**2.a** Verificare se è possibile compilare il progetto

**Risposta**:

Il progetto compila correttamente.

**2.b** Test dell’interfaccia a riga di comando: l’interfaccia funziona correttamente? L’interfaccia è di facile utilizzo? Quali prove sono state eseguite per fare il test?

**Risposta**:

L’interfaccia è chiara, semplice e intuitiva.

Le prove eseguite sono:

* Test “a tappeto” di tutte le funzionalità 🡪 funzionano correttamente
* Forzare scelte al di fuori dell’interfaccia 🡪 gestite bene dal programma

**3 Seconda sessione di test**

**3.a** Definire una procedura di test che permetta di identificare possibili bug nel codice prodotto per le singole classi e i singoli metodi di ogni classe

**Risposta**:

* Operazione preliminare: chiamare tutte le funzioni, quindi i metodi associati ad ogni classe, per verificarne il corretto funzionamento
* Chiamare tutte le funzioni passando argomenti errati: errati, nel senso di “diversi” da come definite nel “.h” corrispondente
* Chiamare la funzione “*Draw*” (definita *virtual* pura), passando argomenti

**3.b** Riportare i risultati del test

**Risposta**:

* Il programma riconosce gli errori volutamente compiuti: restituisce feedback idonei.