**Lorenzo Travi s4980582**

**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.

La documentazione generata permette di orientarsi, tuttavia non compaiono i file “Main(2).cpp” e “polygon.h”.

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.

Il progetto aderisce alle specifiche del progetto in quanto funzioni richieste risultano implementate.

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.

Il file introduce correttamente il progetto e fornisce una breve spiegazione per l’uso dell’interfaccia.

**2 Compilazione e prima sessione di test**

2.a Verificare se è possibile compilare il progetto

La compilazione avviene 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?

L’interfaccia funziona correttamente nella creazione e distruzione di oggetti ed è piuttosto intuitiva.

Il programma è stato utilizzato per creare i poligoni e chiedendone la stampa.

Il programma si blocca quando viene introdotto un valore non intero in tutti i menù di scelta (menù principale, selezione di costruzione, inserimento parametri dell’oggetto in creazione).

Alcune porzioni di interfaccia sono in lingua inglese.

Nel file “Main(2).cpp” del progetto compare diverse volte “goto”. Per una migliore leggibilità del codice si potrebbe sostituire con un ciclo “while”.

**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

Nel file “maintest.cpp” vengono creati e modificati gli oggetti delle diverse classi

3.b Riportare i risultati del test

In RightTrapezoid viene utilizzato in modo equivalente base inferiore e superiore con maggiore e minore.

Le funzioni testate funzionano correttamente.