1. כותרת. מחשבון מטריצות
2. פרטי סטודנט. עקיבא משה סודרנסקי: 326382322, אהרון משה שמסיאן: 205779598
3. הסבר כללי של התרגיל. מאפשר לבצע פעולות שונות על מטריצות, אפשר להוסיף פעולות כמו חיבור, חיסור, כפל בסקלר או טרנספוזיציה, וגם להרכיב פעולות מורכבות שמשלבות כמה פעולות יחד. לאחר מכן, ניתן להפעיל את הפעולות על מטריצות שהמשתמש מזין, או למחוק.
4. רשימת קבצים.  
   squareMatrix – אובייקט מטריצה ריבועית.  
   functionManager – מנהל המחשבון הראשי אחראי כל הקשר בין פונקציות.  
   Operation – אב הפונקציות כולם יורשים הימנו ומאפשר פולימורפיזם.  
   idOperation – פונקצית ID מכפילה במטריצת היחידה.  
   Transpose – פונקצית TRANSPOSE מחליפה שורות ועמודות.  
   Scalar – פונקציית כפל בסקלר.  
   Add – פונקצית המחברת בין שתי תוצאות פונקציות שונות על מטריצות שונות.  
   Sub– פונקצית המחסרת בין שתי תוצאות פונקציות שונות על מטריצות שונות.  
   Comp – פונקציה המפעילה פונקציה אחת ולאחריה מפעילה פונקציה אחרת על אותה מטריצה.  
   OperationTypes, OperationNmaes, OutputConst - קבועים.
5. מבני נתונים עיקרים ותפקידיהם.  
   מטריצה ריבועית – וקטור דו ממדי.  
   וקטור מצביעים מסוג shared\_ptr הנועד לשמור את סוגי הפונקציות לפי הסדר ולאפשר קיום מצביעים לפונקציות מסוימות על אף שמקורם נמחק.
6. אלגוריתמים הראויים לציון.  
   טרנספוז – עוברים רק החלק מעל האלכסון ומעתיקים עם ערך זמני לאינדקס אשר שורתו ועמודתו מוחלפים.
7. תיכון.  
   האובייקט הבסיסי הינו מטריצה ריבועית.  
   קיים אובייקט לניהול המחשבון functionManager.  
   operation הינו אב לכל הפונקציות בכדי לאפשר פולימורפיזם יעיל.  
   id, transpose הינם הבנים הפשוטים כביכול ללא פרמטירים נוספים. אלא רק הפונקציות evaluate, print פולימורפיות.  
   Scalar מכיל גם את הספרה המכפילה אך הוא גם בן פשוט יחסית וevaluate, print פועלות באופן פולימורפי.  
   add, sub, composite הם יותר 'כבדים' יש להם 2 מצביעים מסוג shared\_ptr כי הם עצמם בעצם מחזיקים 2 פונקציות אחרות. גם הם יורשים מoperation וכמובן גם יש להם את הפונקציות הפולימורפיות.
8. לא ידוע.
9. הערות.  
   בפונקציה הפולימורפית evaluate צריך להחזיר מטריצה כי בADD לדוגמא צריך לשמור את הפעולה ולהוסיף לשני, ובכדי למנוע העתקות שלחנו רפרנס ולא החזרנו מטריצה כערך חזרה של הפונקציה.  
   בדוממה לכך העדפנו להשתמש באופרטור += כדי למנוע יצירת מטריצה שלישית ולהעתיק אותה חזרה אז העדפנו לשנות את המטריצה עצמה – מה שדי מסתדר עם התרגיל.