1.1)

Imperative : תכנות אימפרטיבי הינה פרדיגמה אשר מתמקדת בפקודות אשר במהותן משנות את סביבת העבודה (את הstate), בניגוד לתכנות דקלרטיבי שבו מתמקדים ב"מה" לבצע בתפיסה אימפרטיבית מתמקדים יותר ב"איך" לבצע.

Procedural: תכנות פרוצדורלי הינה פרדיגמה שלפיה התוכנית מורכבת מפרוצדורות קטנות, כך שכל אחת מהן עלולה לקרוא לפרוצדורה אחרת בכל זמן נתון בריצת הקוד, תפיסה זאת מגבירה את מודולריות הקוד ומאפשרת להגדיר ממשק ברור למשתמש בתוכנה.

Functional: תכנות פונקציונלי הינה פרדיגמה שלפיה התוכנית מורכבת מפונקציות כאשר כל פונקציה ממפה ערך מסוים לערך אחר, בניגוד לתכנות אימפרטיבי שבו אנו מעדכנים את הstate של התוכנית. התוכנית למעשה הינה הרכבה והפעלה של הפונקציות.

\* יתרונות תכנות פרוצדורלי מול אימפרטיבי –

- מודולריות הקוד – בתכנות פרוצדורלי ניתן להשתמש בפרוצדורה אחת מספר פעמים ולשלוח לה נתונים שונים בכל פעם.

- צמצום הקוד – באמצעות פרוצדורות ושימוש חוזר בהן ניתן לצמצם בצורה משמעותית את כמות הקוד בתוכנית

\* יתרונות תכנות פונקציונלי מול פרוצדורלי –

- תכנות פונקציונלי מאפשר בדיקת קוד ברורה יותר כיוון שהפעלה של פונקציה עם ערך מסוים אינו תלוי בדבר חיצוני כמו בתכנות פרוצדורלי ולכן תמיד יחזיר את אותו הערך עבור אותם פרמטרים.

- סנכרון – בניגוד לתכנות פרוצדורלי אשר שומר על State קיים ומשנה את הState באמצעות פרוצדורות בתכנות פונקציונלי אנו לא מחזיקים state לכן בהרצה מקבילית של הקוד רוב בעיות הסנכרון שנתקלנו בהם בתכנות פרוצדורלי אינן קיימות

1.2) <T> (x : T[], y : (T) => Boolean) => Boolean

)x : Number[]( => Number

<T> (x : Boolean , y : T[] ) => T

1.3) abstraction barriers: זהו תפיסה בתכנות שאומרת שעבור כל קוד שנכתב עליו להיות מחולק ל2 –

צד אחד זהו צד הלקוח, כאשר בצד זה עלינו לסמוך כי התוכנה (חיצונית או אפילו שנכתבה על ידנו) עובדת כראוי ואנו יכולים להשתמש בה כרצוננו בהנחה שהכול עובד טוב.

הצד השני זה צד המממש, בצד זה איננו יכולים להניח כי הקוד עובד אלא להיפך עלינו להניח כי הקוד מכיל באגים ושגיאות ועלינו למצוא ולטפל באותם שגיאות באמצעות טסטים וכו'.

חשוב לומר כי המעבר בין שני הצדדים של ה"מחסום" קורה לעיתים תכופות בעת כתיבת הקוד והמעבר בין שני הצדדים אינו בהכרח בולט, לדוגמה, כאשר נכתוב פונקציה המבצעת דבר כלשהו (צד המממש) אנו לעיתים קרובות נשתמש בפונקציות שכבר מימשנו (צד הלקוח) אך עלינו לבדוק שהפונקציה עובדת כראוי (צד המממש).