חזרה לבחינה

מבנה הבחינה

- הבחינה מתוכננת ל – 180 דקות
- ניתן להשתמש בדף אחד (שני עמודים) שהוכן ע"י הסטודנט
- לבחינה ארבעה חלקים:
  - חלק ראשון חובה (34 נקודות)
  - ויש לבחור שני חלקים מתוך השלושה הנוספים (33 נקודות כל חלק)
  - אם נבחר חלק, יש לענות על כל השאלות שבו
- בכל חלק 4-5 שאלות

על הבחינה

- בסוף כל שאלה מופיע הניקוד שלה
- קרא כל שאלה יותר מפעם אחת, כדי לוודא שהבנת את הנדרש
- לאחר שענית, בדוק שאכן ענית על כל החלקים
- הוסף דוגמאות, כאשר הדבר מתבקש
- כאשר נדרשת השוואה, למשל בין A ל- B, אין הכוונה רק לתאר את A ו – B ולהשאיר לי למצוא את ההבדלים

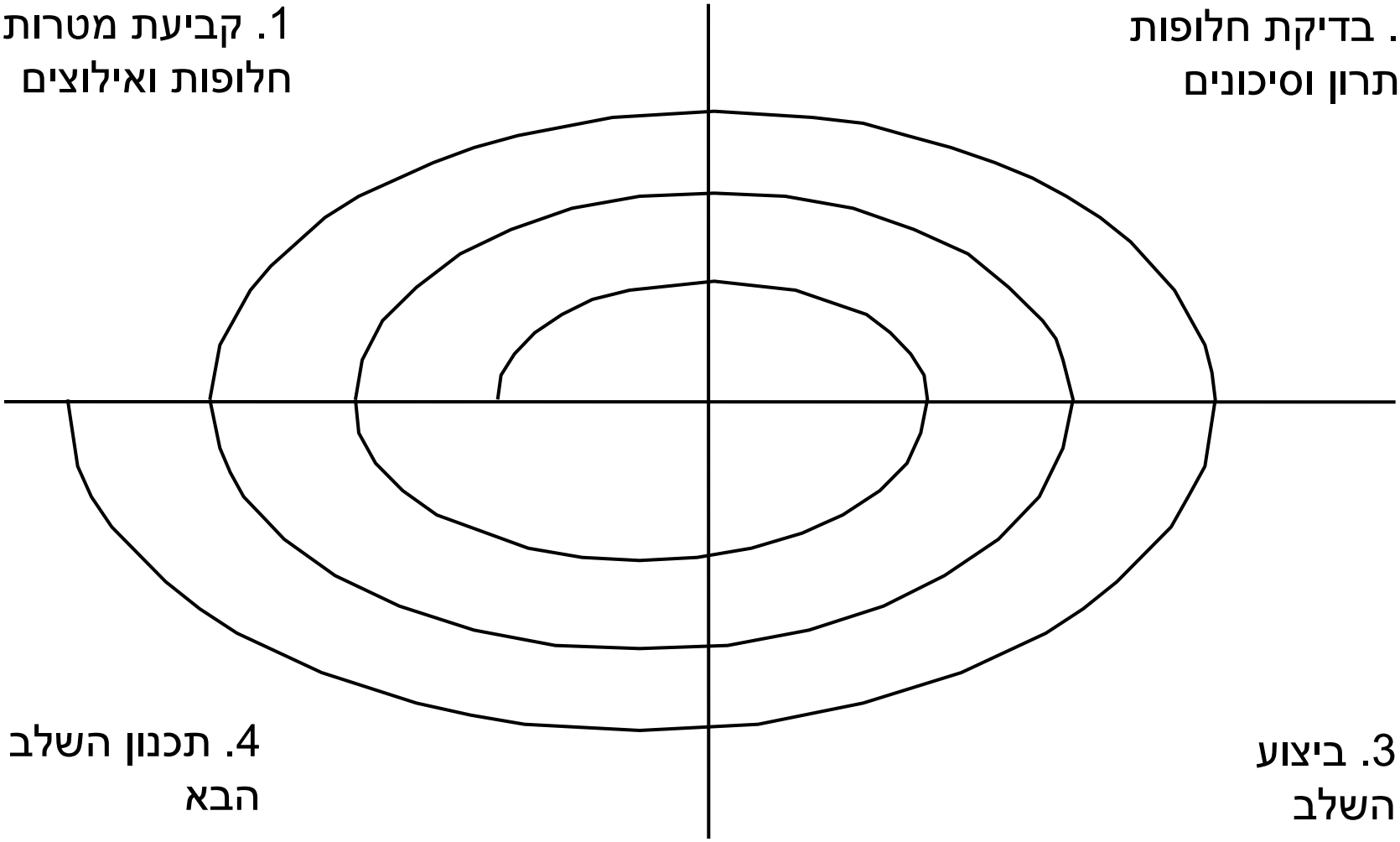
דוגמת שאלה

- מהו מודל הספיראלה? מה לדעתך היתרונות המשמעותיים שלו ומתי כדאי להשתמש בו?

פיתוח ספיראלי

- התהליך מיוצג ע"י ספיראלה במקום רצף פעילויות
- כל לולאה מייצגת שלב בפרויקט
- אין שלבים מובנים כמו דרישות, ניתוח וכד'
- לכל לולאה ניתן להגדיר פעילות בהתאם לצורך
- דגש מיוחד מושם על סיכונים במהלך כל שלב

פיתוח ספיראלי

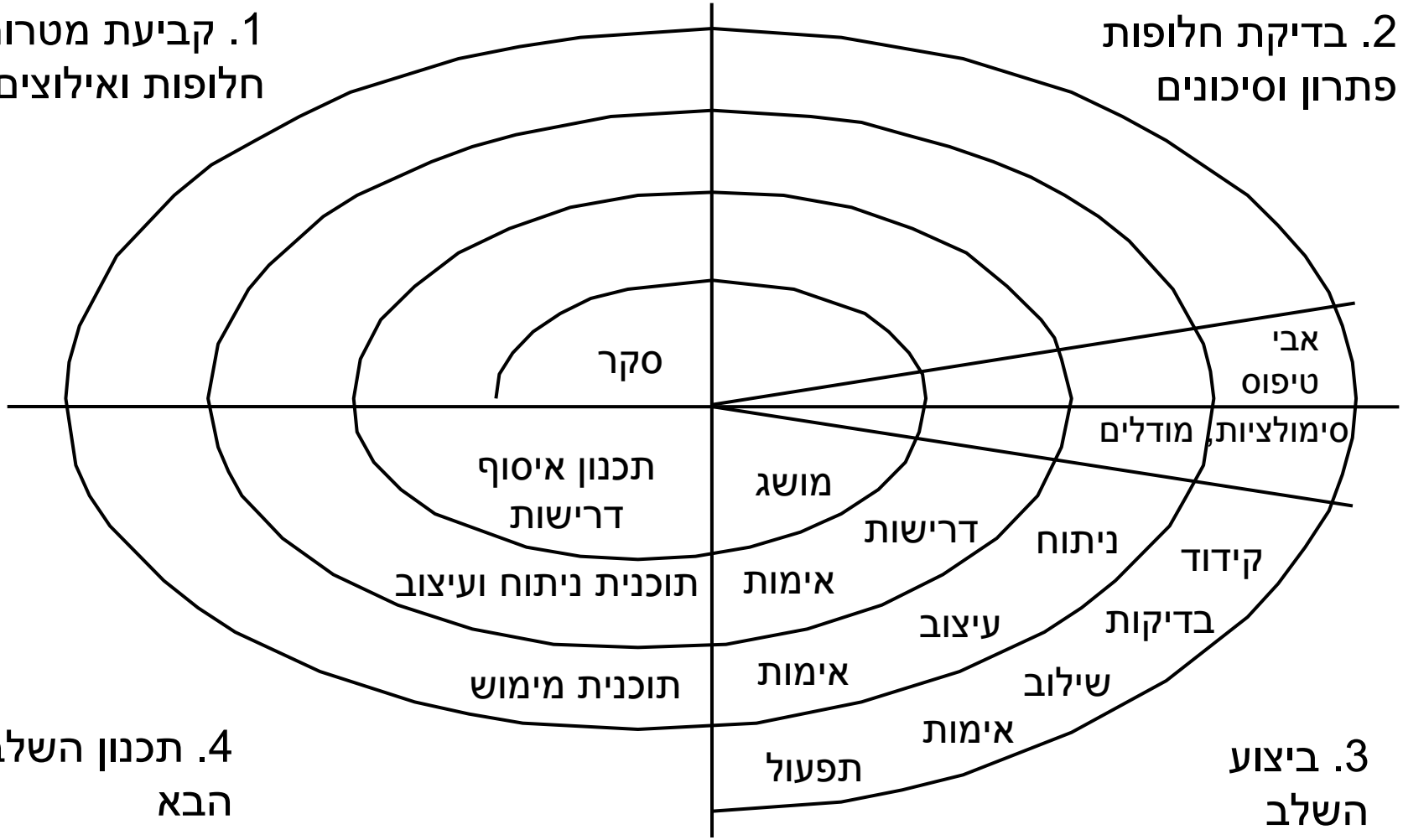


פיתוח ספיראלי

ייזום, דרישות, ניתוח, פיתוח

1. קביעת מטרות חלופות ואילוצים

2. בדיקת חלופות פתרון וסיכונים



3. ביצוע השלב

4. תכנון השלב הבא

הסקטורים בספיראלה

- קביעת המטרות לשלב זה
- ניתוח סיכונים כולל נקיטת אמצעים לצמצומם
- פיתוח ואימות תוך שימוש במודל או סימולציה. המודל יכול להיות כל אחד מהמודלים הכלליים
- תכנון השלב הבא בפרויקט לאחר ניתוח תוצאות השלב הנוכחי

דוגמת שאלה

- הסבר מהן שיטות אג'יליות (Agile) לפיתוח תוכנה, מה העקרונות שלהן ומה היה הגורם שדחף לפיתוחן.

שיטות פיתוח זריזות (Agile)

- התקורה הגבוהה בשיטות העיצוב ה"רגילות" הובילה ליצירה של שיטות חדשות זריזות אשר:
  - שמות דגש על הקוד במקום התכנון/עיצוב
  - מבוססות על תהליך איטרטיבי בפיתוח התוכנה
  - מיועדות לספק תוכנה עובדת במהירות, תוך התייחסות לשינויים בדרישות
- מיועדות במיוחד לפרויקטים קטנים, או לחלקים קטנים של פרויקטים גדולים

שיטות פיתוח זריזות (Agile)

- עקרונות
  - מעורבות מלאה, כל הזמן, של הלקוח בתהליך (הגדרת פונקציונאליות, תעדוף ובדיקת התוצרים המוגמרים)
  - פיתוח אינקרימנטלי
  - דגש על האנשים לא התהליך
  - הבנה שיהיו שינויים ולכן תכנון המערכת כך שתוכל להתמודד איתם בהצלחה
  - שמירה על פשטות. אם רכיב הופך מסובך יש לתכנן אותו מחדש, אולי תוך פיצול לרכיבי משנה

The Agile Manifesto

- העדפת

אנשים ואינטראקציה	על פני	תהליך וכלים
תוכנה עובדת	על פני	תיעוד מפורט
שת"פ עם הלקוח	על פני	מו"מ לחוזה
מענה לשינויים	על פני	הצמדות לתוכנית

Source: www.agilemanifesto.org

דוגמת שאלה

- אחת השיטות הזריזות (Agile)לפיתוח תוכנה היא – Extreme Programming. הסבר את השיטה ואיזה בעיות יכולות להיגרם ממימוש שיטה זו?

Extreme Programming

- השיטה פותחה ע"י Kent Beck כדי לפתור בעיות של מפתחים בסביבה המאופיינת ע"י דרישות לא מוגדרות ומשתנות
- השיטה מבוססת על מספר עקרונות:
  - בחינת קוד תוך כדי הקידוד (קידוד בזוגות)
  - בדיקות יחידה תוך כדי הקידוד
  - עיצוב תוך כדי הפיתוח (Refactoring)
  - שילוב ובדיקות תוך כדי הפיתוח (מספר פעמים ביום)
  - איטרציות קצרות, דקות ושעות ולא שבועות או חודשים

הצלחת פרויקטים כפונקציה של גודל

Size of project	Early	On-Time	Delayed	Cancelled
1 function point	14.68%	83.16%	1.92%	**0.25%**
10 function points	11.08%	81.25%	5.67%	**2.00%**
100 function points	6.06%	74.77%	11.83%	**7.33%**
1,000 function points	1.24%	60.76%	17.67%	**20.33%**
10,000 function points	0.14%	28.00%	23.83%	**48.00%**
100,000 function points	0.00%	13.67%	21.33%	**65.00%**
Average	5.53%	56.94%	13.71%	**23.82%**

Function point ~ 50-100 modern programming language LOC

Patterns of Software Systems Failure and Success, by Capers Jones

עקרונות

- סיפור משתמש ע"י הלקוח
- מפורק למשימות (איטרציות) של עד שבועיים (40 שעות לשבוע)
- הלקוח מגדיר מבדקי קבלה המבוצעים כחלק מהפיתוח
- בסוף המשימה מסופקת מהדורה (חלקית) של המערכת, כאשר בכל איטרציה מתווספת פונקציונאליות
- אם המשימה חורגת משבועיים, מפצלים אותה
- עבודה במקביל ע"י שני מפתחים (בחינת 100% מהקוד)
  - מחקרים הוכיחו שתכנות בזוגות מייצר קוד איכותי יותר ביחידת זמן בהשוואה לשני מפתחים העובדים לבד
- כל אחד יכול לשפר כל דבר

בעיות ב - XP

- בעיות
  - לפעמים קשה לאתר משתמש שיכול לייצג את כל בעלי העניין
  - קשה להוציא אותו מעבדותו ולשמור לאורך זמן על מעורבותו
  - לא כל המפתחים מתאימים לשיטת עבודה כזאת
  - קביעת עדיפויות מוסכמות מסובכת, כאשר יש הרבה בעלי עניין
  - שמירה על פשטות מחייבת עבודה נוספת
  - ייתכן שהחלטות תשתיתיות חשובות אינן נכונות (ארכיטקטורה)
  - לא מתאים לפרויקט במחיר קבוע

דוגמת שאלה

- הסבר את הטבלה להלן

	תכנון	קידוד ובדיקת יחידה	אינטגרציה ובדיקות
מערכת מידע	25%	40%	35%
מערכת web-ית	55%	15%	30%
מערכת זמן אמת	35%	25%	40%
מערכת צבאית	40%	20%	40%

- תכנון של מערכת web–ת הצריך 14 ימים מתי המערכת תעבוד?
- תכנון, קידוד ובדיקות יחידה של מערכת זמן אמת הסתיים ב – 12 חודשים, מתי המערכת תושלם?

זמני "תקן"

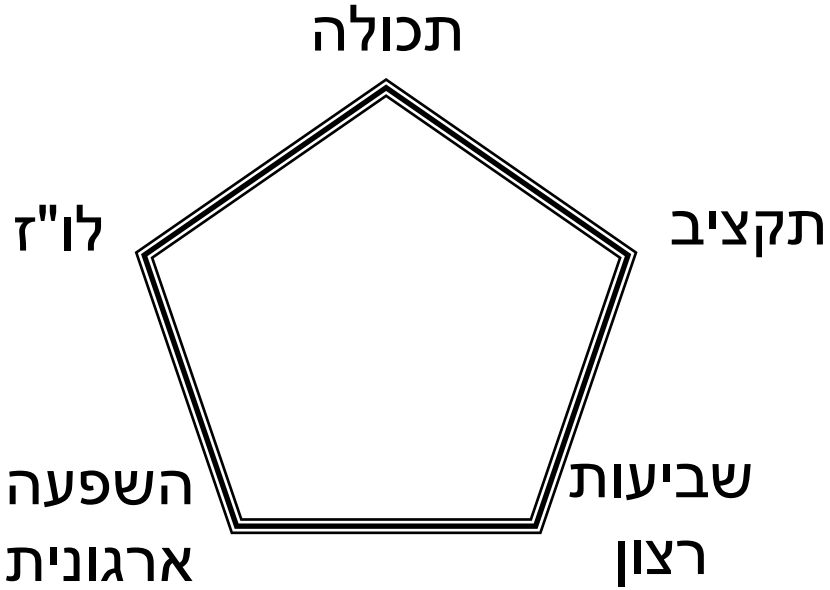
- הטבלה יכולה רק לתת אינדיקציה לגבי הערכות זמנים, אך אין להשתמש בה כתקן מחייב!
- זו יכולה להיות סטטיסטיקה מצטברת לאורך של זמן של ניסיון פיתוח (בארגון למשל), אך אין כל אפשרות להסיק ממנה לגבי פרויקטים אחרים, בלי לבדוק קודם לכן אם הפרויקט החדש דומה ועד כמה למה שהיה בעבר

דוגמת שאלה

- הגדר מהי הנדסת תוכנה למה היא נדרשת

הנדסת תוכנה

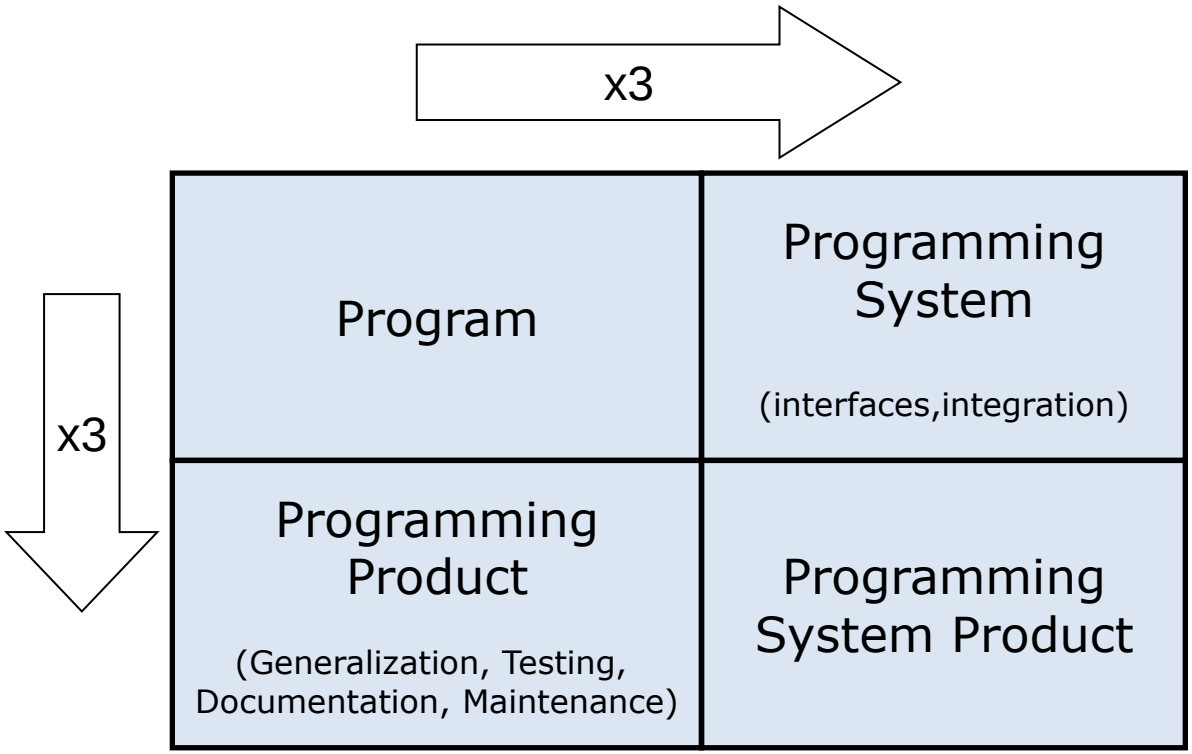
- הנדסת תוכנה הוא תחום ידע שמטרתו להגדיר תהליכים ומתודולוגיות שיאפשרו לפתח תוכנה ללא פגמים, שכולה תסופק בזמן וללא חריגה בעלות
- לפעמים אפשר להוסיף שהתוכנה המסופקת תענה לדרישות המשתמשים



דוגמת שאלה

- התייחס להבדלים שבין פיתוח תוכנית בודדת, פיתוח מערכת, פיתוח מוצר ופיתוח מערכת שהיא מוצר. איך הבדלים אלה מתבטאים בהערכות הזמנים?

מאפיינים



Brooks. F. P. (1975) The Mythical Man-Month

דוגמת שאלה

- נסח מספר המלצות לגבי רישום דרישות שניתנות למדידה, לכל אחד מארבעה המאפיינים: שלמות (Integrity), עקביות (Consistency), סובלנות לשגיאות (Error Tolerance), הדרכה (Training). השתמש בדוגמאות להבהרת תשובתך.

דוגמת תשובה

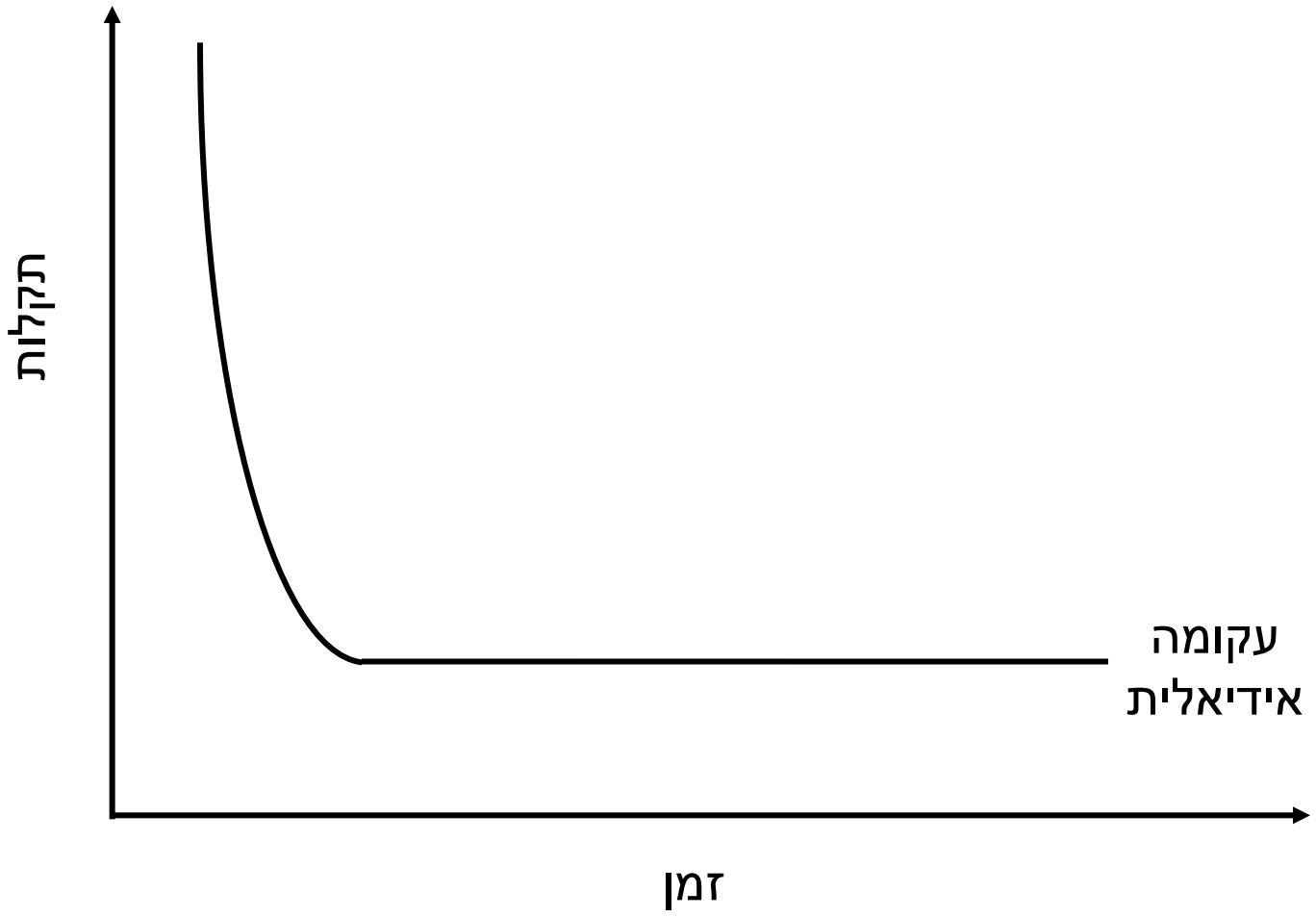
- שלמות (Integrity) – מידת הבקרה המסופקת בגישה למידע או תכניות ע"י גורמים שאינם מורשים
- עקביות (Consistency) – שימוש בעיצוב ותיעוד אחידים במהלך כל פיתוח הפרויקט
- סובלנות לשגיאות (Error Tolerance) – מידת הנזק שנגרמת כאשר התוכנה מגלה שגיאה
- הדרכה (Training) – מידת העזרה הקיימת במערכת ואשר נועדה להדריך משתמשים חדשים

דוגמת שאלה

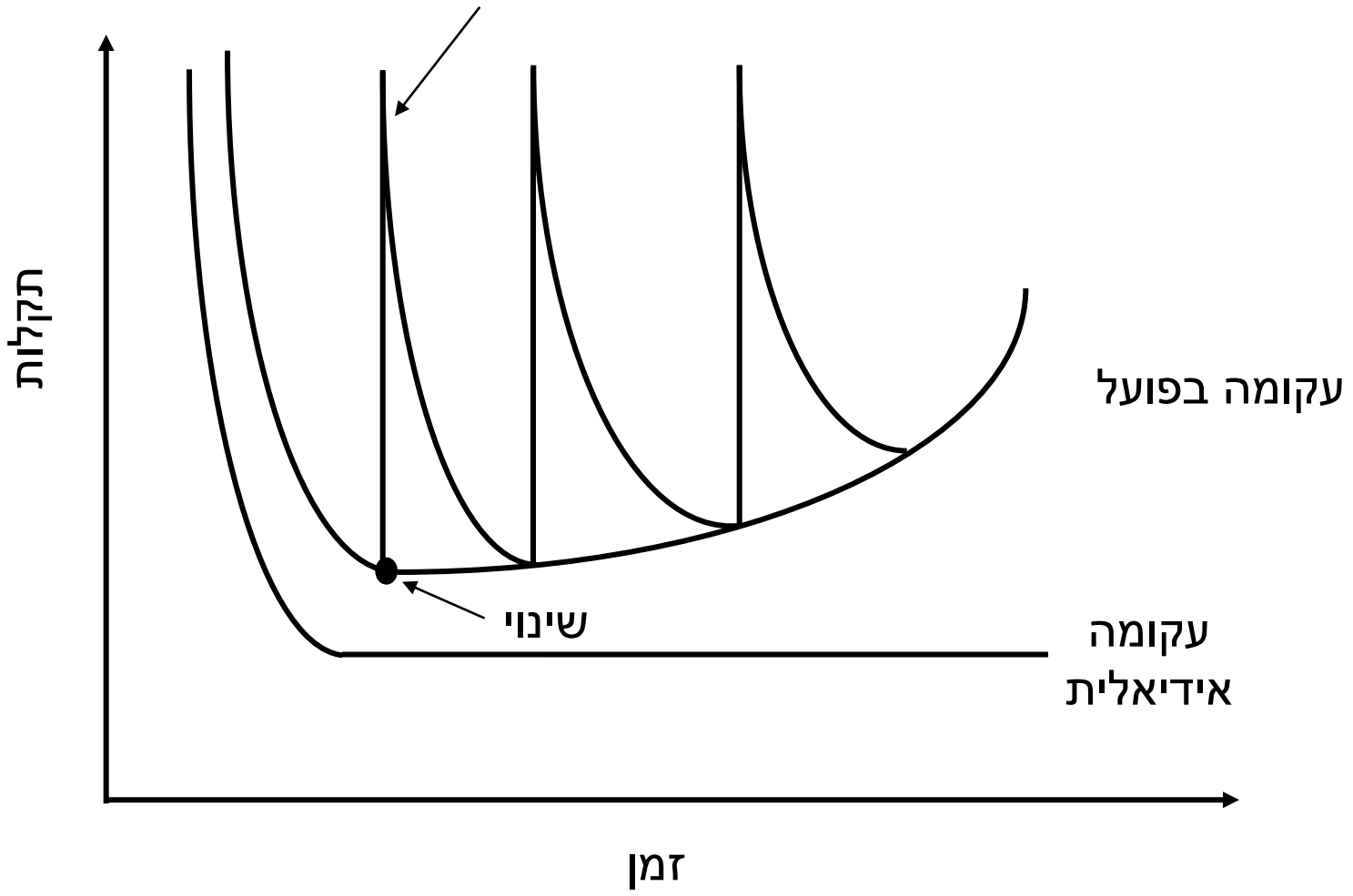
- התייחס לגרף המתאר את מספר התקלות במוצר כפונקציה של זמן והשווה בין תוכנה וחומרה. מה ניתן לעשות על מנת לשפר את המצב?

עקומת חיי מוצר - חומרה

עקומת חיי מוצר – תוכנה (מצב אידיאלי)



עקומת חיי מוצר – תוכנה (בפועל)



חוקי ליהמן (Lehman)

- מאיר ליהמן, מומחה מחשוב אנגלי שעבד מאוחר יותר ב – י.ב.מ הגדיר מספר "חוקים" המתייחסים להתפתחות תוכנה
  - תוכנה חייבת להשתנות ככל שעובר הזמן – יש להיערך לזה
  - כדי לשמור על ארכיטקטורת התוכנה יידרש מאמץ הולך וגדל
- איך לשפר את המצב – הנדסת תוכנה

דוגמת שאלה

- מהו שלב הייזום בפרויקט, מה עושים בו ואיך הוא מסתיים?

ייזום

- התהליך הראשוני שבו מוגדר הבסיס לפרויקט
  - הכנת מסמך הייזום
    - מגדיר ומאשר את שלבי הפרויקט
    - מעניק למנהל הפרויקט את הסמכות לפעול
  - זיהוי בעלי העניין בפרויקט
  - מינוי מנהל לפרויקט

מסמך ייזום

- משמש כאימות השלב
- זיהוי הצורך או הבעיה הארגונית/עסקית
- הגדרת יעדי הפרויקט ומטרותיו
- הגדרת הדרישות למערכת החדשה (מוצרים, שירותים, תוצאות)
- בדיקת כדאיות (ROI)
- הגדרת מדדי הצלחה
- איתור המשתמשים
- בחינת מספר חלופות והחלטה על החלופה העדיפה
- הגדרה ראשונית של היקף הפרויקט ותכולתו

מסמך ייזום (המשך)

- צורך עסקי
  - דרישות השוק (משתמשי המוצר המפותח)
  - צורך ארגוני (הרחבת יכולות למשל לצורך שיווק)
  - דרישת לקוח (או משתמשי הלקוח)
  - התקדמות טכנולוגית (מימוש טכנולוגיה חדשה)
  - אילוצים חוקיים (שינויי חקיקה או עדכון תקנות)
  - דרישות חברתיות (שיפור נגישות)

דוגמת שאלה

- התייחס לדמיון ולשוני שבין פרויקטים הנדסיים (תכנון וייצור מוצר הנדסי/מוחשי כלשהו) לבין פיתוח תוכנה

פרויקטים הנדסיים ותוכנה - דמיון

- מיקוד טכני/הנדסי
- נתיב קריטי
- בדיקות
- ניהול משאבי אנוש
- ניהול משימות
- דיווח התקדמות
- טכניקות איכות

פרויקטים הנדסיים ותוכנה - שוני

- תוכנה אינה מוגדרת בצורה טובה
- הרבה תהליכי הפשטה (אבסטרקציה)
- עקומת לימוד תחום הפעילות הספציפי
- בעיות בהגדרת ממשקים
- זמינות משאבי ידע
- חוסר בתקנים
- חוסר ידע לגבי עלויות ותמחור

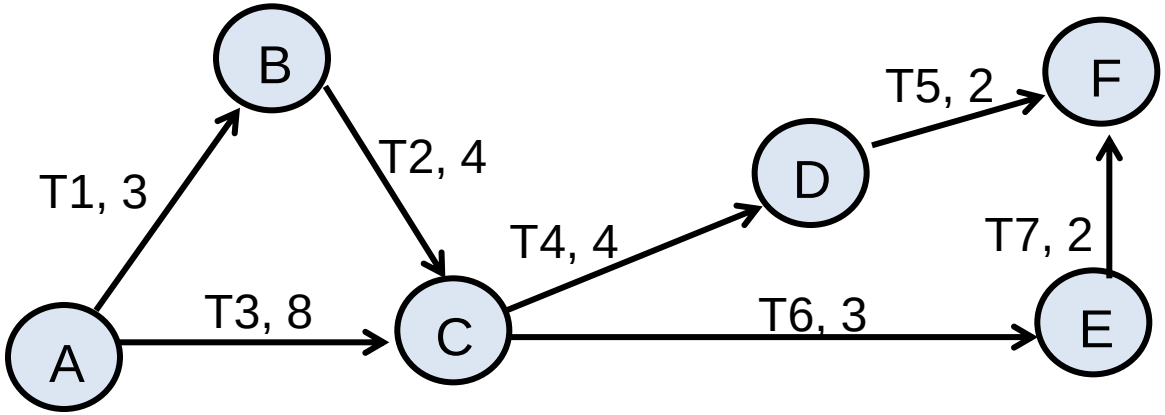
דוגמת שאלה

- מהו נתיב קריטי ומה חשיבותו לפרויקט? שרטט את רשת הפעילות של הפרויקט המתואר וחשב את הנתיב הקריטי

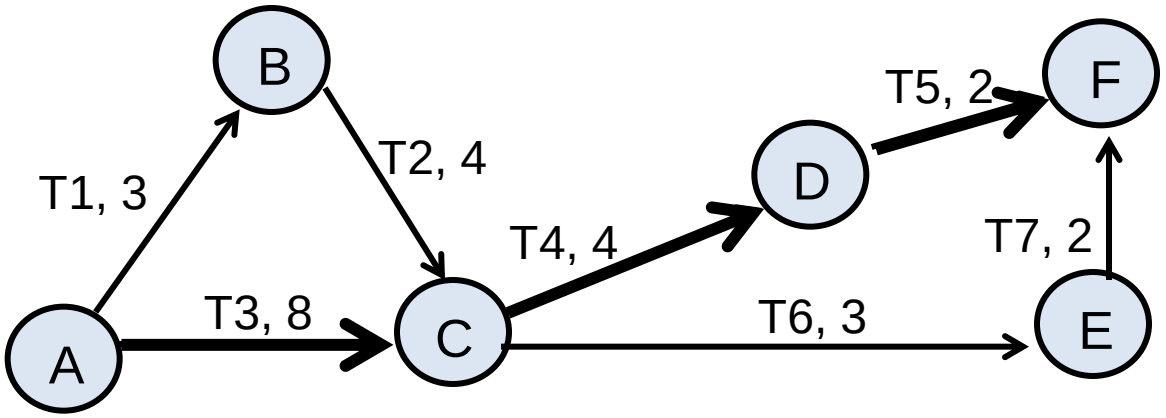
Task	Dependent On	Duration
T1	-	3
T2	T1	4
T3	-	8
T4	T2,T3	4
T5	T4	2
T6	T2,T3	3
T7	T6	2

דוגמת תשובה

Task	Dependent On	Duration
T1	-	3
T2	T1	4
T3	-	8
T4	T2,T3	4
T5	T4	2
T6	T2,T3	3
T7	T6	2



דוגמת תשובה (נתיב קריטי)

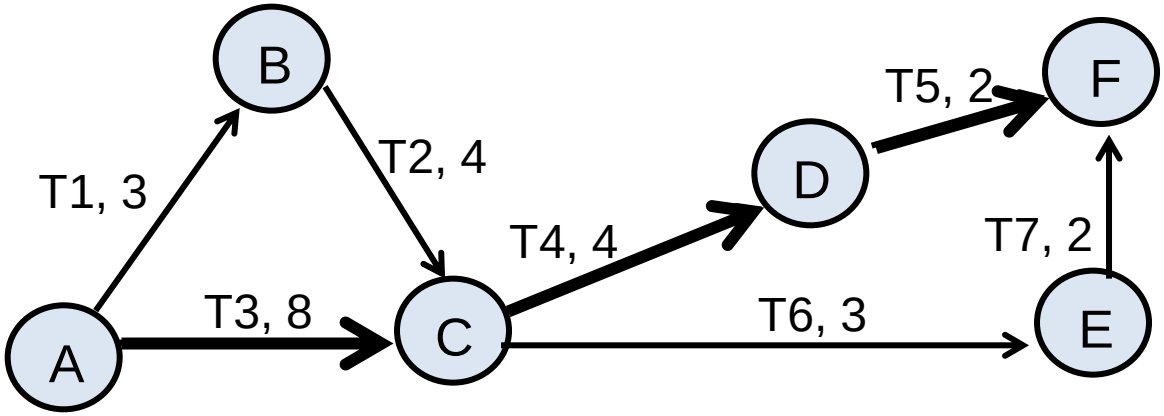


Path	Time
T1, T2, T4, T5	13
T1, T2, T6, T7	12
T3, T4, T5	14
T3, T6, T7	13

Critical Path: T3, T4, T5

המשך השאלה

- מה קורה אם פעילות T1 מתעכבת ביום אחד?
- מה קורה אם היא מתעכבת ביומיים?
- מה קורה אם פעילות T6 מתעכבת ביום אחד?

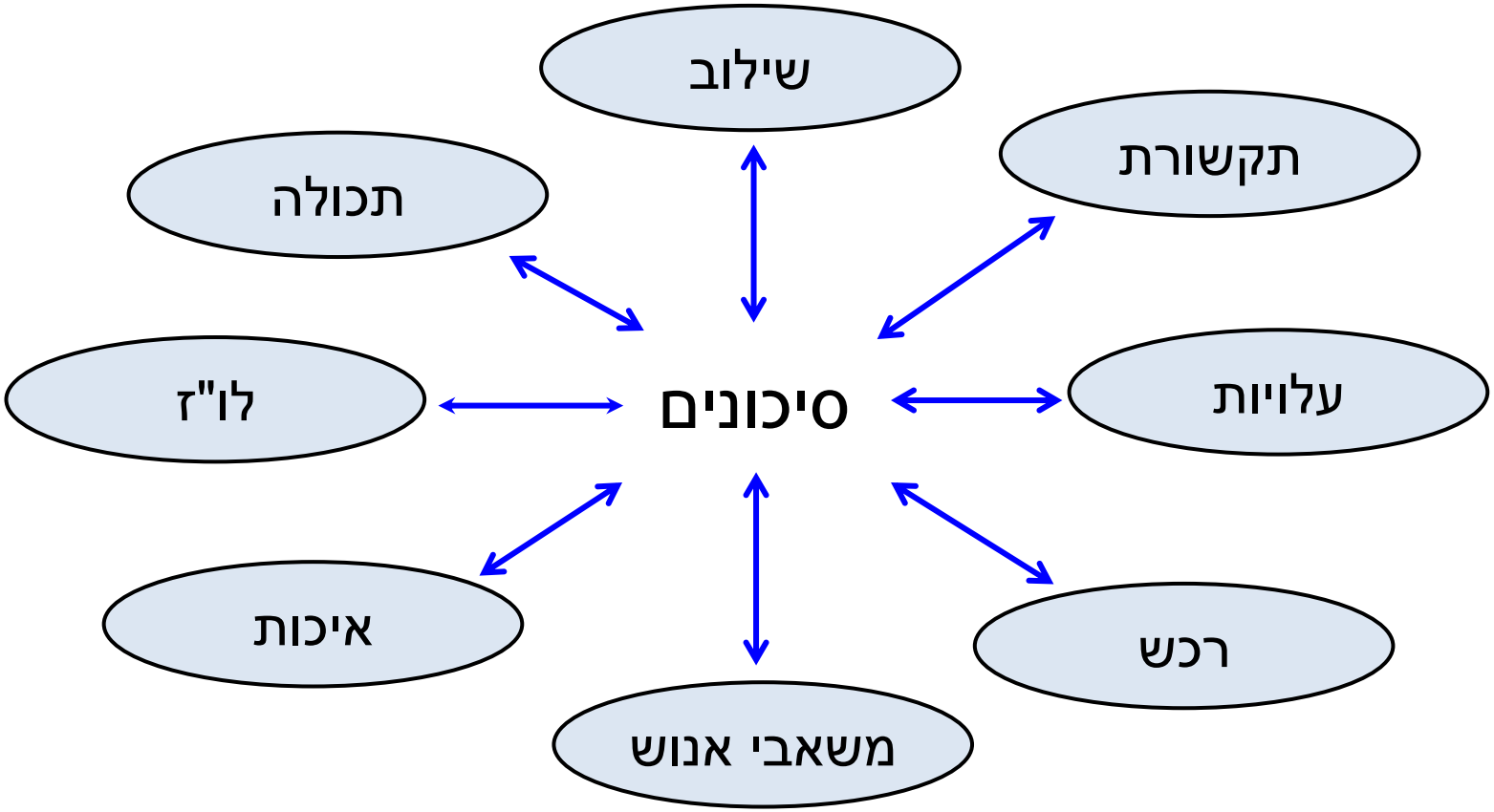


דוגמת שאלה

- בשעה טובה סיימת את לימודך בהצלחה. לאור ניסיונך הרב (שאת חלקו צברת במהלך ההרצאות) בפיתוח אפליקציות לטלפונים חכמים, קיבלת פרויקט לפיתוח מספר אפליקציות עסקיות עבור לקוח תעשייתי. הנקודות החשובות ביותר ללקוח הן עמידה בזמנים שתוכננו ואיכות המוצר המפותח. מאחר ופיתחת כבר מספר יישומים דומים, נראה שלא תהיה לך כל בעיה עם הדרישות. למרות זאת, לדעתך, איזה גורמי סיכון יכולים לצוץ?

מהו סיכון?

- סיכון הוא כל אירוע, פעילות או גורם אשר מפריע לביצוע התקין של הפרויקט ומונע את סיומו המוצלח



גורמי סיכון

- הלקוח/השוק
  - בעיות כלכליות אצל הלקוח/בשוק, אי הבנת הדרישות
- הארגון המפתח
  - בעיות משאבים
- הפרויקט
  - בעיות בתכנון
- המוצר/הסביבה
  - בעיות בכלים, סביבת הפיתוח
- קבלני משנה/ספקים
  - בעיות באספקות

דוגמת שאלה

- הסבר מהי דרישה למערכת? התייחס גם לסוגי דרישות

מהי דרישה?

- הגדרת הדרישה יכולה לנוע מתיאור כללי מופשט (אבסטרקטי), ועד תיאור מפורט (לדוגמה נוסחה מתמטית)
- דרישה יכולה גם לשמש לשתי מטרות:
  - בסיס לקבלת הצעות – ולכן מוגדרת בצורה כללית
  - בסיס לחוזה – ולכן מפורטת
- בשני המקרים מדובר על דרישות

הגדרת דרישות (Davis)

"אם ארגון רוצה לפתח מערכת בעזרת קבלנים חיצוניים, הוא חייב להגדיר את הדרישות בצורה כללית, ומבלי לכפות את הפתרון.

את הדרישות יש להגדיר בצורה שתאפשר לקבלנים שונים, דרכי מימוש שונות.

לאחר שנקבע הזוכה ומנסחים את החוזה, יש להגדיר את הדרישות בצורה מפורטת, אשר תאפשר למשתמש להבין, ולאמת מה המערכת תספק לו.

שני מסמכים אלה, נקראים מסמכי דרישות למערכת"

סוגי דרישות

- דרישות המשתמש – נכתבות בשפה טבעית, לעיתים בתוספת תרשימי הבהרה, ומגדירות את שירותי ואילוצי המערכת
- דרישות המערכת – מסמך מובנה המגדיר בפירוט את המערכת. מצורף לחוזה בין הצדדים
- מפרט המערכת – תיאור מפורט של המערכת, שנועד לצורך מימושה

סיווג דרישות

- דרישות פונקציונאליות – השירותים שהמערכת צריכה לספק. (התנהגות במצבים מסוימים)
- דרישות שאינן פונקציונאליות – אילוצים המשפיעים על שירותי המערכת
- דרישות "סביבתיות" (נסתרות) - דרישות המוכתבות ע"י תחום הפעילות בו המערכת עובדת (נגזרות מאילוצי התחום)

דוגמת שאלה

- הסבר את ההבדל בין בדיקות תוכנה (Testing) לבין ניפוי (Debugging). מה הכישורים שנדרשים לבצע כל אחת מהפעילויות?

בדיקות

- בדיקות (Testing) נועדו להדגים שהמערכת עובדת (או לא)
- ניפוי (Debugging) הוא תהליך של איתור התקלה (באג) ותיקונו
- בדרך כלל קודם מבצעים בדיקות ובעקבותיהן, אם נמצאו תקלות, יבוצע הניפוי

בדיקות

- בהיבט הפסיכולוגי, בדיקות מוכיחות קיום בעיה והניפוי מתקן אותה
- בעוד שבדיקות אפשר לבצע ללא הכרה מעמיקה, ניפוי מחייב הכרה לעומק של הקוד
- בדיקה יכולה להיעשות ע"י גורמים חיצוניים וחלקים ממנה יכולים להיות אוטומטיים

דוגמת שאלה

- לאור הצלחתך בלימודים ומומחיותך בתחום פיתוח יישומים לטלפונים חכמים, זכית במכרז לפיתוח יישום מפות "פשוט"
- לאחר הפעלת היישום המשתמש מקליד כתובת יעד והיישום שלך אמור לשרטט את הנתיב אל היעד, לרשום במלל את שלבי הנסיעה ולהוסיף את משך הזמן הנדרש
- חשוב על מספר נקודות להבהרה שנדרשות לך בטרם תתחיל לתכנן ולפתח

דוגמת תשובה

- לאיזה סוגי מכשירים זה אמור להתאים?
- האם נדרשת אפשרות לבחירת נקודת המוצא?
- האם נדרשות אופציות לקביעת המסלול?
  - למשל:
    - הדרך הקצרה ביותר
    - המהירה ביותר
    - הזולה (ללא כבישי אגרה)
    - שילוב אופציות?

דוגמת תשובה

- האם אפשר להגדיר נקודה (אחת או יותר) בדרך שיש לעבור בה?
- נדרש גם מסלול הפוך?
- האם יש אפשרות להגדיר את מועד הנסיעה? עכשיו? בלילה?
- האם אפשר לקבוע שהכוונה להליכה ברגל?
- האם יש להוסיף שכבות מידע?
  - למשל חניונים באזור היעד
  - עבודות בכביש
- ...

סוף החזרה לבחינה