שעת קבלה - עבודה spl 1

לא בודקים על יעלות, לא צריך מימוש מטורף אבל גם לא לחפור בשביל פעולות פשוטות

באופן כללי על העבודה -

תוכנית שיקום עיר

מורכבת מכמה אלמנטים וכל תוכנית נותנת תרומה שונה.

מכיוון שזאת סימולציה נוכל ליצור מספר תוכניות שיקום לאותו יישוב כדי לשקול אסטרגיות שונות לכל תוכנית.

אפשר (וצריך) להוסיף מתודות ומשתנים אבל לא להוריד מתודות ומשתנים קיימים, לוודא שכל מה שקיבלנו בקובץ אנחנו באמת מממשים.

על הקובץ -

הmain הוא זה שיתחיל את הפרויקט

בהרצת התוכנית צריך לתת לה את הנתיב לconfig_file שנמצא בתיקייה של הפרויקט וזה בעצם הארגומנט שנקבל בmain

סימולציה -

מתחזקת את רשימת היישובים, תוכניות, פעולות שבוצעו וכל הפסיליטיז שאפשר לממש בהמשך לפי הקובץ קונפיגורציה.

הלולאה של התוכנית תעבוד כל עוד isrunning הוא true ונדרש לשנות לfalse כשאנחנו לא רוצים להמשיך לקלוט דברים מהמשתמש

סטלמנט (יישוב) -

לכל יישוב יש שם ייחודי, מסוג מסוים כקובע כמה מבנים הוא יכול לבנות ברגע נתון (מיוצג בenum) בעבודה עם סוג היישוב אפשר לקרוא להם בשם או בערך המספרי שמייצג אותם. חייב לבנות את המקסימום של המבנים שאפשר ברגע נתון

סלקשן פוליסי (אבסטרקטית) -

מחלקות המימוש -

המחלקה הראשונה

כל אובייקט כזה מקבל רשימה של הפסיליטיז שעומדים לרשותה. נעבור על הרשימה לפי האינדקסים. בפעם הראשונה שנקרא למתודה select נרצה להחזיר את האיבר הראשון ובפעם השניה את האיבר השני וכו. כדי לדעת איפה היינו בפעם האחרונה יש משתנה lastSelectedIndex.

המחלקה השניה

כל מתקן תורם בצורה שונה לתוכנית שיקום.

המשתנה של המרחק מחושב לפי ההפרש בין המספר המקסימלי והמינימלי לאחר שחישבנו את הscore של כל נושא. נרצה לבחור את הפסיליטיז שיוצרים מרחק הכי קטן. במקרה שיש 2 פסיליטיז אופטימלים שיוצרים אותו מרחק נבחר לפי סדר ההופעה

המחלקה השלישית

תבחר פסיליטיז שרק מקטגורית economy ובאופן דומה מחזירה את הפסיליטיז לפי הבחירה האחרונה. לדוג אם יש ברשימה 5 פסיליטיז אבל רק 2 ו-4 מסוג economy אז בפעם הראשונה של הקריאה למתודה נחזיר 2 ובפעם השניה את פסיליטי 4 ופעם השלישית נחזור לתחילת הרשימה ונבחר שוב את 2. ונעדכן את הlastSelectedIndex

המחלקה הרביעית

באותו קונספט אבל עם קטגוריה של סביבה

פסילטיייי (מתקן) -

לכל מתקן יש כמה קטגוריות שמאפיינות אותו (facilityType)

המתקן יכול להיות חלק מכמה תוכניות שונות לכן שייך למחלקה פסיליטי שמחזיקה מידע נוסף עליו שנוכל לזהות אותו ולנהל אותו (לדוג אם הוא בבנייה או מוכן)

*יש טעות בקובץ, זה יהיה לפי השם של הסטלמנט ולא לפי id

כשיוצרים את הקונפיגורציה אנחנו לא יוצרים פעולה addsetelment ואז עושים act, אלה בונים את הישוב לבד ואז מוסיפים אותו לרשימה שמתחזקת את היישובים

הפעולות נועדו רק לאינטרקציה עם המשתמש.

אנחנו לא צריכים את הaction הזה, אנחנו פשוט צריכים ליצור instance ונשים לב שאנחנו גם לא רוצים לתעד את הפעולה הזאת כי זה המצב ההתחלתי של הסימולציה

יצירת תוכנית plan -

ביצירה צריך לשלוח את המדיניות בחירה

לכל תוכנית יש סטטוס שאומר אם אפשר לבנות פסיליטיז או לא, כלומר האם התוכנית זמינה או לא.

סיום התוכנית -

לשחרר את הזיכרון של כל הפעולות בתוכניות בheap

גיבוי -

המשתנה חיצוני למחלקות ומאותחל בnull

הגיבוי (לאחר בקשת המשתמשת) לוקח את המשתנה ומבצע השמה עם copyConsrucator

ובריסטול מחזיר את מה ששמור בגיבוי להיות המצב העיקרי שלנו

צעד בסימולציה -

בודקת אם יש אפשרות לבנות עם מבנים (לפי הזמינות של התוכניות הקיימות)

אם כן ממלא את התוכנית שיקום עד שאין מקום יותר.

מכאן נבור על כל המבנים שהבניה שלהם עוד לא הסתיימה. הtime left של פסיליטי הוא בעצם העלות שלו ונפחית את הזמן שנותר להם לסיום הבניה ב1. כשנגיע ל0 נעדכן אותו מהרשימה של הפסיליטיז שתחת בניה לאלה שמוכנים (אופריישנל)

(יש עוד שלבים לפי הסכמה המתוארת) - כל הסכמה מייצגת מה קורה בצעד בתוכנית

שינוי של הסטטוס של מבנה קורה רק במסגרת הסכמה הזאת

דיבאג -

קודם להגדיר את הbuild (לחזור להקלטה)

יש תוסף שקוראים לו runner - זה שואל איזה קובנפיגורציה להפעיל, בוחרים לפי מה שהגדרנו אחרי הbuild

איך להתחיל את העבודה?

- לגשת לmain ולנסות להריץ
- לשים בהערה את מה שצריך ולוודא שהסביבה מוכנה
- לשים בהערה את מה שקשור לסימולציה
- להתחיל לממש מחלקות קטנות נגיד סטלמנט שהם לא תלויות במחלקות אחרות
- אחרכ לממש מחלקות מורכבות כמו פסיליטיז (גם לא תלויה אבל יותר מורכבת)
- ורק אז מחלקות שתלויות אחד בשני
- כל הפעולות עם הaction וכו לממש אחרון כי זה גורר מתודות שמשנות את הstate של הסימולציה

דגשים חשובים -

- לשים לב בשימוש של act רק באינטרקציה עם המשתמש
- לא למחוק/לשנות חתימה של פונקציה ומשתנים
- יש מימוש שעוזר לפרסר את הקלט מהמשתמש