

پکیج sample شامل :

کلاس main که در آن گرافیک پروژه و لاجیک اصلی برنامه قرار دارد
کلاس مرج سورت که به آن آرایه ای از دیوار ها داده میشود و بر اساس مختصات ایکس آنها
مرج سورت میشوند
کلاس table که مشابه کلاس موزاییک تبیل در پروژه دیگر است و محیط رسم آسمانخراش هارا
می سازد
و کلاس wall که دارای یک مختصات ایکس و یک ارتفاع که ایگرگ نامیده شده و یک تایپ می
باشد که مشخص میسازد از چپ به راست دیوار شروع یا پایانی یک ساختمان است

پس از اجرای متد main متد start اجرا میشود
یک نمونه از table ساخته میشود و اضلاع آن را به سازنده آن تابع میدهیم بسته به ارتفاع و
مختصات ایکس دور ترین ساختمان سپس در start
متد run را صدا زده می شود که لاجیک اصلی برنامه در آن قرار دارد
در متد run ,

ابتدا ورودی های برنامه را از طریق کنسول میگیرد به همان فرم داده شده در صورت سوال
این کار توسط متد input انجام گرفته و آرایه ای از دیوار ها برگردانده می شود
هر آسمان خراش داده شده به دو دیوار شروع و پایان تقسیم میشوند که دارای ارتفاع یکسان و
مختصات شروع و پایان آن آسمانخراش هستند
سپس یک نمونه از کلاس مرج سورت ساخته میشود و دیوار هارا به آن میدهیم تا آنها را مرتب
کند بر اساس مختصات ایکس از کوچک به بزرگ یعنی دیوار ها از چپ به راست
سپس دیوار های مرتب شده را به وسیله تابع getSortedWalls میگیریم و در sortedWalls
ذخیره میکنیم
برای کشیدن این آسمان خراش ها نیاز به یک سری نقاط داریم که از این دیوار های مرتب شده
آنها را بدست می آوریم و در انتها به شکل اصلی اضافه میکنیم
برای ساختن نقاط دیوار های مرتب شده را به متد createPoints میدهیم

این متد آرایه ای از نقاط می سازد برای ذخیره سازی و یک صف اولویت برای تشخیص بلند ترین ساختمان حاضر می سازد

ابتدا در این صف مقدار صفر را قرار میدهیم چون هیچ ساختمانی وجود ندارد سپس از سمت چپ ترین دیوار شروع میکنیم و سه سمت راست میرویم

- اگر دیواری که به آن رسیده ایم دیوار شروع یک آسمان خراش باشد آنرا با ارتفاعی که در آن هستیم یعنی بالا ترین ارتفاع در صف اولویت مقایسه میکنیم اگر کوچکتر مساوی بود لازم نیست حرکتی کنیم یا نقطه ای بسازیم اما اگر بزرگتر بود نقطه برخورد ارتفاع فعلی و ساختمان جدید و همچنین نقطه شروع ساختمان جدید با ارتفاعش را وارد آرایه نقاط میکنیم

توجه داشته باشید که در مرحله اول چون بر روی زمین قرار داریم و بیشترین مقدار صف اولویت صفر است نقاط پایین و بالای دیوار شروع اولین آسمانخراش اضافه میشوند پس از اضافه کردن نقاط ارتفاع آن آسمانخراش را به صف اولویت اضافه میکنیم

- اگر دیواری که به آن رسیده ایم دیوار پایان یک آسمانخراش باشد آنرا با صف اولویت مقایسه میکنیم اگر برابر بیشترین مقدار آن بود یعنی پایان آسمان خراشی است که بالاترین ارتفاع را دارد بنابراین دو نقطه پایانی آن ساختمان با ارتفاعش و نقطه برخورد آسمانخراش زیرش با آن دیوار اگر دو بعدی نگاه کنیم به آرایه نقاط اضافه میکنیم و سپس آن ارتفاع را از صف اولویت حذف میکنیم.

در غیر این صورت هم لازم نیست کاری کنیم ارتفاع آسمانخراشی که به دیوار پایانی آن رسیده ایم را در صف اولویت حذف میکنیم

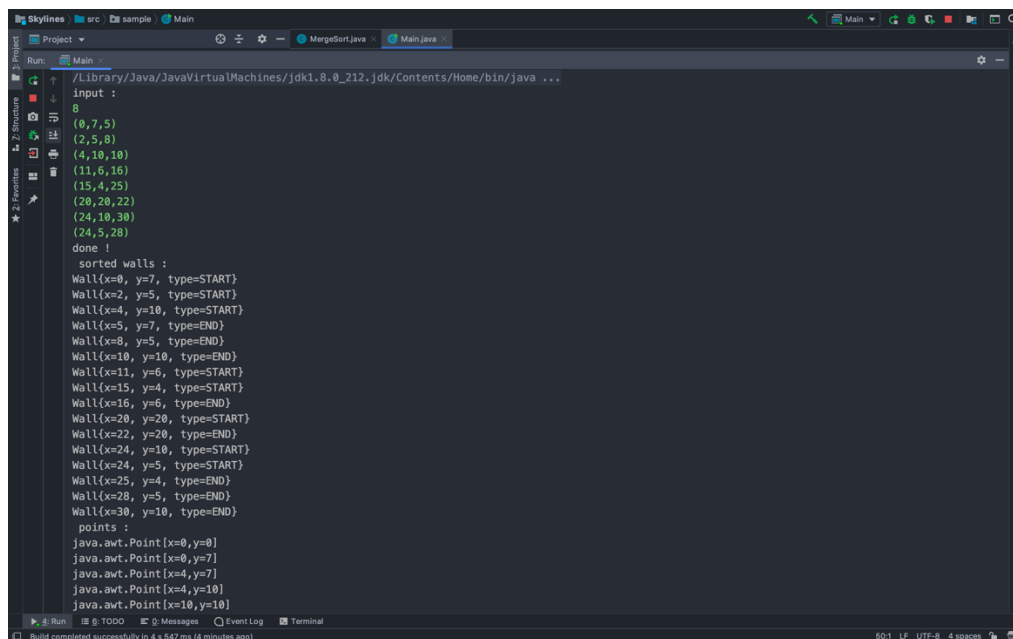
این کار را آنقدر ادامه میدهیم تا به پایان همه آسمانخراش ها برسیم سپس آرایه نقاط را بر میگردانیم

در متد run پس از این مرحله این نقاط را به متد drawLines موجود در کلاس Table میدهیم و هر دو نقطه کنار هم را با یک خط به هم وصل کرده و و نمایش میدهیم

در شکل خروجی آسمانخراش ها بدون همپوشانی و بصورت دو بعدی مشاهده می شوند

پیچیدگی زمانی برنامه مربوط به بخش مرج سورت میباشد که $2 * n$ عدد را مرتب میکند بنابراین پیچیدگی زمانی آن $n \lg n$ می باشد.

اجرای برنامه :



```
Run: Main
/Library/Java/JavaVirtualMachines/jdk1.8.0_212.jdk/Contents/Home/bin/java ...
Input :
0
(0,7,5)
(2,5,8)
(4,10,10)
(11,6,16)
(15,4,25)
(20,20,22)
(24,10,30)
(24,5,28)
done !
sorted walls :
Wall{x=0, y=7, type=START}
Wall{x=2, y=5, type=START}
Wall{x=4, y=10, type=START}
Wall{x=5, y=7, type=END}
Wall{x=8, y=5, type=END}
Wall{x=10, y=10, type=END}
Wall{x=11, y=6, type=START}
Wall{x=15, y=4, type=START}
Wall{x=16, y=6, type=END}
Wall{x=20, y=20, type=START}
Wall{x=22, y=20, type=END}
Wall{x=24, y=10, type=END}
Wall{x=24, y=5, type=START}
Wall{x=25, y=4, type=END}
Wall{x=28, y=5, type=END}
Wall{x=30, y=10, type=END}
points :
java.awt.Point[x=0,y=0]
java.awt.Point[x=0,y=7]
java.awt.Point[x=4,y=7]
java.awt.Point[x=4,y=10]
java.awt.Point[x=10,y=10]
```

ورودی برنامه با هشت آسمانخراش

