



**زیربرنامه:**

Find(T address, type tname)

|  |  |  |
| --- | --- | --- |
| **توسعه دهندگان** | مرتضی نامور |  |
| **تهیه کنندگان مستند** | مرتضی نامور | |
| **تاییدکنندگان** | مرتضی نامور | |
| **تاریخ تنظیم سند** | 22/02/1394 | |
| **شناسه سند** | **MC2F003F1** | |
| **زبان برنامه‌نویسی** | **Fortran 90** | |

1. وظایف

همانطور که قبلاً در مورد ساختار داده ای platform توضیح داده شد ، به منظور سهولت در فرآیند به روزرسانی اطلاعات شبکه ، تعدادی از اطلاعات را در اشیاء شبکه ذخیره نمی کنیم و برای دستیابی به آن ها باید در میان اطلاعات موجود جستجو کرده و آن ها را بیابیم.

در این platform این وظیفه بر عهده متد find است که برای کاربر ، اطلاعاتی را که به طور مستقیم قابل دسترسی نیستند ، پیدا کرده و بر اساس نوع آن ها در اختیار کاربر قرار دهد.

اطلاعاتی که در این platform به صورت مستقیم قابل دسترسی نیستند عبارتند از :

• گره های یک المان (زیرا هر المان فقط وجوه خود را ذخیره می کند)

• المان های همسایه یک وجه

• المان های همسایه یک المان

* وجوه متصل به یک گره
* المان های متصل به یک ضلع

این پنج گروه اطلاعات فوق را متد find می تواند برای کاربر پیدا کند.

1. توضیحات و تئوری­ها

با توجه به اینکه متد find باید بتواند انواع مختلفی از اطلاعات و متغیر ها را به عنوان ورودی بپذیرد و به عنوان خروجی برگرداند لذا به مانند متد A2N باید از template استفاده بشود.

متد find دارای دو ورودی می باشد . ورودی اول آدرس یا به عبارتی اشاره گر (pointer) یک شئ ( گره، وجه، المان و ضلع) و ورودی دوم متغیری از نوع TYPE (پارامتر TYPE یک متغیر از نوع enum است که در ابتدای کلاس MESH قرار گرفته است) می باشد که مشخص می کند کاربر چه چیزی را از متد find به عنوان خروجی می خواهد. به طور مثال اگر ورودی آدرس یک المان باشد ، کاربر باید مشخص کند که آیا گره های آن المان را می خواهد یا المان های همسایه آن المان را یا اگر ورودی آدرس یک ضلع باشد طبیعتاً کاربر به دنبال پیدا کردن المان های متصل به آن ضلع است. نکته ای که در مورد این متد وجود دارد این است که این متد اطلاعاتی را که پیدا می کند ، با توجه به نوع آن ها درون متغیر های global به نام های Point ، Face و Cell قرار می دهد. خروجی خود متد از نوع int بوده که مشخص کننده تعداد اشیائی است که در هر حالت پیدا کرده است.

1. بخش­های زیربرنامه
2. تعریف متغیرهای اولیه

بدون توضیح

1. ذخیره آدرس ورودی به متد

در این بخش آدرس ورودی در قالب reinterperet\_cast در متغیری به نام add از کلاس union ذخیره می شود. سپس یک آرایه به نام s تعریف می شود که درایه های آن تفاضل آدرس خانه اول حافظه اشغال شده توسط اشیا و آدرس اشغال شده در متغیر add توسط اشیاء متناظر آن می باشد. به ترتیب درایه اول مربوط به المان ها ، درایه دوم مربوط به اضلاع و درایه سوم مربوط به گره هاست.

1. شروع فرآیند پیدا کردن گره های یک المان

در این بخش این شرط بررسی می شود که اگر درایه اول آرایه s بزرگتر از 0 و کوچکتر مساوی تعداد المان های شبکه باشد و همچنین متغیر type ورودی برابر با point باشد آن گاه متد شروع به پیدا کردن گره های المان ورودی به متد می کند.

سپس متغیرهای لازم در ادامه این بخش تعریف می شود. به طور مثال P\_counter تعداد گره های پیدا شده است.

1. پیدا کردن گره های المان های از نوع تتراهدرال

در این بخش ابتدا بررسی می شود که اگر المان ورودی از نوع تتراهدرال بود آنگاه ابتدا سه گره اولین وجه این المان درون آرایه Point ذخیره می شود. این کار به کمک دستور memcpy صورت می گیرد. سپس در بین گره های وجه دوم المان جستجو می شود و آن گرهی که مخالف سه گره قبلی باشد پیدا می شود. در انتها چهار گره المان ورودی در آرایه Point قرار گرفته اند.

1. پیدا کردن گره های المان های از نوع مکعبی (cube)

در این بخش ابتدا بررسی می شود که اگر المان ورودی از نوع مکعبی باشد آن گاه ابتدا چهار گره وجه اول آن (وجه با اندیس 0 ) درون آرایه Point کپی می شود. سپس چهار گره وجه دوم آن (وجه با اندیس 1 ) درون آرایه Point کپی می شود. این نکته را باید مدنظر داشت که وجه اول و دوم یک المان مکعبی وجوه مقابل به یکدیگر هستند و هیچ گره مشترکی با هم ندارند.

1. پیدا کردن گره های المان های منشوری (Prism)

در این بخش ابتدا بررسی می شود که اگر المان ورودی از نوع منشوری باشد آنگاه سه گره وجه اول آن (وجه با اندیس 0) در آرایه Point کپی شود. سپس سه گره وجه دوم آن (وجه با اندیس 1) در آرایه Point کپی می شود. این نکته حائز اهمیت است که وجوه المان های منشوری به گونه مرتب شده اند که وجه اول و دوم آن ، وجوه سه گرهی مقابل به یکدیگر هستند.

1. پیدا کردن گره های المان های هرمی با قاعده مربع (pyramid)

در این بخش ابتدا بررسی می شود اگر المان ورودی از نوع هرم قاعده مربع باشد آنگاه چهار گره وجه اول آن (وجه با اندیس 0 ) درون آرایه Point کپی می شود . سپس در بین گره های وجه دوم آن جستجو می شود و هر گرهی که مخالف چهار گره پیدا شده قبلی باشد ، درون آرایه Point کپی می شود. وجوه یک المان هرم به گونه ای مرتب شده است که وجه اول آن ، وجه چهار گرهی باشد.

1. شروع فرآیند پیدا کردن المان های متصل به یک وجه

در این بخش بررسی می شود که اگر ورودی های متد find به صورتی باشند که باید المان های متصل به یک وجه پیدا شوند آن گاه متغیر های اولیه مربوطه را تعریف می کنیم. سپس با جستجو در بین المان های متصل به گره اول (گره با اندیس 0) وجه ورودی به متد ، آن المان هایی را می یابیم که این وجه ، یکی از وجوه آن المان باشد. این المان ها را درون آرایه Cell کپی می کنیم. باید توجه داشته باشیم که تعداد این المان ها نمی تواند از دوتا بیشتر باشد.

1. شروع فرآیند پیدا کردن المان های متصل به یک المان

در این بخش کافیست المان های متصل به هر وجه المان ورودی را پیدا کنیم. سپس آن المان هایی را که برابر با خود المان ورودی نباشند را درون آرایه Cell کپی کنیم. به همین منظور ابتدا بر روی وجوه المان ورودی حرکت کرده و با فراخوانی متد find المان های متصل به آن وجه را پیدا می کنیم. در صورتیکه تعداد المان های متصل به آن وجه برابر با دو باشد آن گاه آن المانی را که برابر با المان ورودی نیست را درون آرایه C قرار می دهیم. در صورتیکه تعداد المان های متصل به هر وجه برابر با یک باشد آن المان قطعا همان المان ورودی است لذا نباید درون آرایه C قرار بگیرد. در انتها تمامی درایه های آرایه C را درون آرایه Cell قرار می دهیم.

1. شروع فرآیند پیدا کردن وجوه متصل به یک گره

در این بخش ابتدا المان های متصل به گره را فراخوانی می کنیم. بدین منظور آدرس آرایه ای که این المان ها در آن ذخیره شده اند را درون متغیر c می ریزیم. سپس در بین وجوه اولین المان متصل به گره ورودی جستجو می کنیم و هر وجهی که یکی از گره های آن برابر با گره ورودی باشد را درون آرایه Face قرار می دهیم. مشابه این کار را با نوشتن یک حلقه بر روی مابقی المان ها و جستجو در بین تک تک وجوه آن ها انجام می دهیم. در اینجا برای اینکه وجوه تکراری درون آرایه Face قرار نگیرد از متد check\_in استفاده می شود.

1. شروع فرآیند پیدا کردن المان های متصل به یک ضلع

با توجه به اینکه در ساختمان داده ای پلتفرم، اضلاع موجود در شبکه هندسی درون آرایه ای مجزا ماندد گره ها، وجوه و المان ها ذخیره نمی شوند لذا در موارد خاصی ممکن است در یک برنامه نیاز باشد بین دو گره ضلعی تشکیل شود و سپس المان های متصل به آن جستجو شود. بدین منظور کافیست المان های متصل به دو گره این ضلع در دو آرایه ذخیره شود و تعداد آن ها نیز در دو متغیر قرار گیرد. سپس در صورتیکه تعداد المان متصل به گره اول از المان های متصل به گره دوم بیشتر باشد آنگاه در بین المان های متصل به گره اول جستجو می کنیم و تمام گره های آن را به کمک متد find پیدا می کنیم. سپس بررسی می کنیم که اگر یکی از گره هر کدام از آن المان ها با گره دوم ضلع ورودی به متد برابر باشد آنگاه آن المان متصل به ضلع ورودی به متد خواهد بود. در صورتیکه تعداد المان های متصل به گره دوم بیشتر باشد فرآیند فوق به صورت برعکس انجام خواهد گرفت . یعنی در بین المان های متصل به گره دوم جستجو می کنیم و با کمک متد find گره های هر کدام را پیدا می کنیم. سپس در صورتیکه یکی از گره های آن المان ها برابر با گره اول ضلع ورودی باشد آن گاه آن المان به ضلع ورودی متصل خواهد بود.