ليلحت اتيادمتعاالا يف اقاج

لامعتلل عم ،كارسفتسا دلأعس صنلا يجمربلا مدقملا ةدالولل طخلخ ةيعبتلا فوفصل □□□□، ثيح متي عيمجت اتيعبتلا سبحب ةيمرهـ ددحم ىوتسم نم ةيمرهـ. زحلا ىلع ليبس، لاثملا اذإ كان level=1، متي عيمجت زحلا في ىوتسملا ىلعألا ىلع□ ليبس لاثملا، □□□□□□ كان وإذا level=2، متي اهعيمجت في ىوتسملا يناثلا ىلع□ ليبس، لاثملا □□□□ □□□□□□

```
    Args:
        file_path (str):        .java.

    Returns:
        list:                   .
    """
    imports = []
    try:
        with open(file_path, 'r', encoding='utf-8') as f:
            for line in f:
                match = re.search(r'^\s*import\s+([\w.]+);', line)
                if match:
                    imp = match.group(1)
                    #                 )         import java.util.*;)
                    if not imp.endswith('.*'):
                        imports.append(imp)
    except Exception as e:
        print(f"   :          }file_path}: {e}")
    return imports

def get_package_group(full_class_name, level):
    """
                        'level'          .

    Args:
        full_class_name (str):             )         "org.springframework.boot.App").
        level (int):                        )          1    "org"  2    "org.springframework").

    Returns:
        str:           )         "org"    "org.springframework").
    """
    package = '.'.join(full_class_name.split('.')[:-1])  #
    parts = package.split('.')
    if len(parts) <= level:
        return package  #                                        level
    else:
        return '.'.join(parts[:level])  #         'level'

if __name__ == '__main__':
    #                 :     root_directory   level
```

```
if len(sys.argv) != 3:
    print(:    " python script.py <root_directory> <level>")
    sys.exit(1)

root_dir = sys.argv[1]
try:
    level = int(sys.argv[2])
    if level < 1:
        raise ValueError
except ValueError:
    print(:  "         level       "  (
    sys.exit(1)

all_classes = set()

#     :
for root, dirs, files in os.walk(root_dir):
    for file in files:
        if file.endswith('.java'):
            file_path = os.path.join(root, file)
            package = get_package(file_path)
            if package:
                class_name = file.replace('.java', '')
                full_class_name = f"{package}.{class_name}"
                all_classes.add(full_class_name)

#
group_dependencies = set()

#     :
for root, dirs, files in os.walk(root_dir):
    for file in files:
        if file.endswith('.java'):
            file_path = os.path.join(root, file)
            package = get_package(file_path)
            if package:
                class_name = file.replace('.java', '')
                full_class_name = f"{package}.{class_name}"
                importer_group = get_package_group(full_class_name, level)
                imports = get_specific_imports(file_path)
```

```
                for imp in imports:
                    #
                    #
                    if imp in all_classes and imp != full_class_name:
                        imported_group = get_package_group(imp, level)
                        if imported_group != importer_group:
                            group_dependencies.add((importer_group, imported_group))


# DOT
print('digraph G {')
for from_group, to_group in sorted(group_dependencies):
    print(f'  "{from_group}" -> "{to_group}";')
print('}')
```

الاتغييرات شرح

1. تحديث الوسائط المطلوبة:
   - الأصلية: python script.py <root_directory>
   - المعدلة: python script.py <root_directory> <level>
   - أضيف دعم للحجة الثانية، level، التي تحدد مستوى الحزم المراد تحليلها. يجب أن تكون الحجة من نوع صحيح. يتم التحقق من وجود level الحجة وتأكيد أن level عدد صحيح موجب.
2. دالة جديدة: get_package_group:
   - تمت إضافة دالة لحساب مجموعة الحزمة لملف معين بناءً على level المحدد.
   - يستخرج الحزمة من اسم الملف الكامل على سبيل المثال، [illegible]، ويأخذ أجزاء [illegible] إلى أجزاء: [illegible]، [illegible]، [illegible] أول level أجزاء:
     - إذا كان level=1: يعيد [illegible].
     - إذا كان level=2: يعيد [illegible].
     - إذا كانت الحزمة لها أجزاء أقل من level على سبيل المثال، [illegible] مع level=3، يعيد الحزمة الكاملة [illegible].
3. تجميع التبعيات:
   - الأصلية: استخدم defaultdict(set) لتخزين التبعيات بين الملفات.
   - المعدلة: استخدم set group_dependencies لتخزين العلاقات بين المجموعات الحزمة لإزالة التكرارات (from_group, to_group).
   - لكل ملف:
     - احسب مجموعة importer_group باستخدام get_package_group.
     - لكل استيراد في الملف imp in all_classes إذا كان imp != full_class_name:
       - احسب مجموعة الملف المستورد imported_group.

- إذا كانت مجموعات الاستيراد مختلفة (`imported_group != importer_group`)، فيضيف إلى `group_dependencies`.
- يضمن set، لذا الفردية، تؤدي التبعيات المتعددة بين نفس المجموعات إلى حافة واحدة.

4. **إخراج**:
   - **الأصلية**: طباعة الحواف بين الوصفوف الفردية على سبيل المثال، `[[1, 2], [3, 4], [5, 6], [7, 8]]`.
   - **المدعلة**: طباعة الحواف بين مجموعات مزحلة على سبيل المثال، `[[1, 2], [3, 4]]` لـ `level=2`.
   - إخراج على للحواف مرتبة للوصول على الحواف.

متطلبات تحقيق كيفية

- **دعم التبعيات**: تحليل جميع الحزم قبل تجميع level الآن يقبل النص البرمجي الذي.
- **المستوى 1**: على سبيل المثال، على سبيل المثال، `[[1, 2], [3, 4], [5, 6], [7, 8]]`. على سبيل المثال، `[[1, 2], [3, 4]]` و`[[5, 6], [7, 8]]` هما في مجموعة `[[1, 2], [3, 4]]`، لذا لا داخل `[[5, 6], [7, 8]]` تظهر بين هما الاستيرادات داخل الحواف.
- **المستوى 2**: على سبيل المثال، `[[1, 2], [3, 4], [5, 6], [7, 8]]`. على سبيل المثال، `[[1, 2], [3, 4]]` إلى `[[1, 2], [3, 4], [5, 6], [7, 8]]` من استيراد المثال، `[[1, 2], [3, 4], [5, 6], [7, 8]]` إلى `[[1, 2], [3, 4]]`.
- **التبعيات تحليل قبل مجموعة الحزم**: تحديد النص البرمجي لكل صف ءانب على level قبل تحليل الفردية. الصفوف وليس المجموعات، بين التبعيات أن يضمن مما استيراداته،
- **المثال مع التوافق**: لمثال الحافة:
  - الأصلية: `"org.springframework.boot.web.servlet.server.Session" -> "org.springframework.boot.convert.Du`
  - مع `level=2`: هما في كل `[[1, 2], [3, 4], [5, 6], [7, 8]]`، لذا لا فيضيف حافة مجموعة `[[1, 2], [3, 4]]`. واحدة.
  - مع `level=3`: `[[1, 2], [3, 4], [5, 6], [7, 8]]` `[[1, 2], [3, 4], [5, 6], [7, 8]]`، فيضيف حافة بين هذه المجموعات. المختلفة.

استخدام أمثلة

- **الأمر**: `python script.py /path/to/project 1`
  - المجموعات: `[[1, 2], [3, 4], [5, 6], [7, 8]]`, `[[1, 2], [3, 4], [5, 6], [7, 8]]`، إلخ.
  - الإخراج: يتضمن قد `"org" -> "com"` إذا استورد حزمة في صف حزمة في صف `[[1, 2], [3, 4], [5, 6], [7, 8]]`.
- **الأمر**: `python script.py /path/to/project 2`
  - المجموعات: `[[1, 2], [3, 4], [5, 6], [7, 8]]`, `[[1, 2], [3, 4], [5, 6], [7, 8]]`، إلخ.
  - الإخراج: يتضمن قد `"org.springframework" -> "org.apache"`.

الملاحظات

- `all_classes` داخل استيرادات فقط المشروع داخل التبعيات تتضمن من الأصلي سلوكه على البرمجي النص يظل الذاتية. والتبعيات البعد استيرادات واستبعاد

- متي لهاجت فوفصلا نودب الصاوم عم صنلا يجمربلا. الأصلي
- تاجرإلا وه طخم هجوم في غةيص □□□، نكمي هتدهاشم مدختسا بأدوات مثل □□□□□□□□□.

يجب أن يفي هذا النص الجمربلا دعملا لماكلا بكلطب عيمجتلا تايعبتلا بسحو ىوتسم ةيمره. الحزم