# Code:

**Classes.py:**

```
###
#  вариант В задания варианта 22
###
class Lang:
    """language"""

    def __init__(self, id, lang_type, title):
        self.id = id
        self.lang_type = lang_type
        self.title = title


class Lib:
    """library"""

    def __init__(self, id, title, methods_count, lang_id):
        self.id = id
        self.title = title
        self.lang_id = lang_id
        self.methods_count = methods_count


class LibLang:
    """lang library"""

    def __init__(self, lang_id, lib_id):
        self.lang_id = lang_id
        self.lib_id = lib_id


langs = [
    Lang(1, 1, "java"),
    Lang(2, 2, "javaScript"),
    Lang(3, 1, "c++"),
    Lang(4, 3, "c"),
    Lang(5, 2, "f#")
]

libs = [
    Lib(1, "jakarta", 12000, 1),
    Lib(2, "tanzorFlow", 300, 2),
    Lib(3, "sfml", 234, 3),
    Lib(4, "gson", 1234, 5),
    Lib(5, "jackson", 12345, 1),
    Lib(6, "c lib", 12, 4)
]

lib_lang = [
    LibLang(1, 4),
    LibLang(2, 1),
    LibLang(3, 2),
    LibLang(4, 1),
    LibLang(1, 2),
    LibLang(5, 5),
    LibLang(2, 5),
```

```
    LibLang(3, 3),
    LibLang(2, 4)
]


def main():
    """Основная функция"""
    one_to_many = {}
    for la in langs:
        one_to_many[la.title] = [(li.title, li.methods_count) for li in libs
if li.lang_id == la.id]

    first(one_to_many)
    second(one_to_many)

    many_to_many_temp = [(la.title, li.title)
                         for la in langs
                         for li in libs
                         for lali in lib_lang
                         if lali.lang_id == la.id and lali.lib_id == li.id]
    third(many_to_many_temp)

# задание 1
def first(one_to_many):
    print("№1:")
    result = dict(filter(lambda item: len(item[1]) > 0,
                         {key: list(filter(lambda item:
str(item[0]).startswith("j"), val)) for key, val
                          in one_to_many.items()}.items()))
    print(result)


# задание 2
def second(one_to_many):
    print("№2:")
    result = sorted({k: min([val[1] for val in one_to_many[k]]) for k in
one_to_many.keys()}.items(),
                    key=lambda item: item[1])
    print(result)


# задание 3
def third(many_to_many):
    print("№3:")
    result = sorted(many_to_many, key=lambda item: item[1])
    print(result)


if __name__ == '__main__':
    main()
```

## Результат выполнения:

№1:

{'java': [('jakarta', 12000), ('jackson', 12345)]}

№2:

[('c', 12), ('c++', 234), ('javaScript', 300), ('f#', 1234), ('java', 12000)]

№3:

[('java', 'tanzorFlow'), ('java', 'gson'), ('javaScript', 'jakarta'), ('javaScript', 'gson'), ('javaScript', 'jackson'), ('c++', 'tanzorFlow'), ('c++', 'sfml'), ('c', 'jakarta'), ('f#', 'jackson')]

[('java', 'gson'), ('javaScript', 'gson'), ('javaScript', 'jackson'), ('f#', 'jackson'), ('javaScript', 'jakarta'), ('c', 'jakarta'), ('c++', 'sfml'), ('java', 'tanzorFlow'), ('c++', 'tanzorFlow')]

Process finished with exit code 0