Министерство науки и высшего образования РФ
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Уфимский государственный авиационный технический университет»

Кафедра Высокопроизводительных вычислительных технологий и систем

	1	2	3	4	5	6	7	8	9	10
100										
90										
80										
70										
60										
50										
40										
30										
20										
10										
0										

ИЗУЧЕНИЕ ЧИСЛЕННОГО ПАКЕТА OPENFOAM

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе по дисциплине

«Дифференциальные уравнения»

1507.334111.000 ПЗ

Группа ПМ-253	Фамилия И.О.	Подпись	Дата	Оценка
Студент	Шамаев И.Р.			
Консультант	Михайленко К.И.			
Принял	Лукащук В.О.			

Уфа 2021

Федеральное государственное бюджетное образовательное учреждение высшего образования
«Уфимский государственный авиационный технический университет»

Кафедра Высокопроизводительных вычислительных технологий и систем

ЗАДАНИЕ

на курсовую работу по дисциплине

«Дифференциальные уравнения»

Студент: Шамаев Ильдар Рустемович Группа: ПМ-253

Консультант: Михайленко Константин Иванович

1. Тема курсовой работы

Численное решение дифференциальных уравнений механики сплошной среды средствами OpenFOAM

2. Основное содержание

2.1. Изучить основу работы в OpenFOAM.

2.2. Генерация сетки в утилите Blockmesh.

2.3. Структура решателя и его запуск.

2.4. Постобработка результата.

2.5. Оформить пояснительную записку к курсовой работе.

3. Требования к оформлению материалов работы

Требования к оформлению пояснительной записки

Пояснительная записка к курсовой работе должна быть оформлена в соответствии с требованиями ГОСТ и содержать

- титульный лист,
- задание на курсовую работу,
- содержание,
- введение,
- заключение,
- список литературы,
- приложение, содержащее листинг разработанной программы, если таковая имеется.

Дата выдачи задания Дата окончания работы

"__" _____ 202_ г. "__" _____ 202_ г.

Консультант _____ Михайленко К.И.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

OpenFOAM — свободно распространяемый инструментарий вычислительной гидродинамики для операций с полями. В частности, пакет позволяет решать задачи гидродинамики ньютоновских и неньютоновских вязких жидкостей, как в несжимаемом, так и сжимаемом приближении с учётом конвективного теплообмена и действием сил гравитации. Возможно решение дозвуковых, околозвуковых и сверхзвуковых задач.

В основе пакета лежит набор библиотек, предоставляющих инструменты для решения систем дифференциальных уравнений в частных производных, как в пространстве, так и во времени. Рабочим языком кода является ООП C++. В терминах данного языка большинство математических дифференциальных и тензорных операторов в программном коде уравнений может быть представлено в удобочитаемой форме, а метод дискретизации и решения для каждого оператора может быть выбран уже пользователем в процессе расчёта. Таким образом, в коде полностью инкапсулируются и разделяются понятия расчетной сетки (метод дискретизации), дискретизации основных уравнений и методов решения алгебраических уравнений.

Цель исследования – освоение современного открытого численного пакета OpenFOAM. Цель достигается путем решения следующих задач:

1. Исследование алгоритма Simple
2. Исследование алгоритма Piso
3. Реализация алгоритмов средствами OpenFOAM

1. Математическая модель несжимаемой жидкости

Уравнения Навье-Стокса для однофазного течения с постоянной плотностью и вязкостью имеют следующий вид:

$$\frac{\partial U}{\partial t} + \nabla \cdot (\vec{v}\vec{v}) - \nabla \cdot (\nu \nabla \vec{v}) = -\nabla p$$

Для несжимаемой жидкости уравнения Навье–Стокса следует дополнить уравнением несжимаемости:

$$\nabla \cdot \left(\rho \vec{U}\right) = 0$$

Решение этой пары уравнений не является простым, поскольку явное уравнение для давления недоступно. Один из наиболее распространенных подходов состоит в том, чтобы вывести уравнение давления, взяв дивергенцию уравнения импульса и подставив его в уравнение непрерывности.

2. Уравнение решения динамики жидкости в пакете OpenFOAM

Уравнение импульса может быть переписано следующим образом:

$$a_p\vec{U_p} = H(\vec{U}) - \nabla p \Longleftrightarrow \vec{U_p} = \frac{H(\vec{U})}{a_p} - \frac{\nabla p}{a_p}$$

где

$$H(\vec{U}) = -\sum_n a_n \vec{U}_n + \frac{\vec{U}^o}{\Delta t}$$

Первый член $H(\vec{U})$представляет собой матричные коэффициенты соседних ячеек, умноженные на их скорость, в то время как вторая часть содержит нестационарный член и все источники, кроме градиента давления.

Уравнение непрерывности дискретизируется следующим образом:

$$\nabla \cdot \vec{U} = \sum_f \vec{S} \cdot \vec{U}_f = 0$$

где $\vec{S}$- направленный наружу вектор площади грани и $\vec{U}_f$скорость на грани.

Скорость на поверхности получается путем интерполяции полудискретизированной формы уравнения импульса следующим образом:

$$\vec{U}_f = \left(\frac{H(\vec{U})}{a_p}\right)_f - \frac{(\nabla p)_f}{(a_p)_f}$$

Подставляя это уравнение в полученное выше дискретное уравнение неразрывности, получаем уравнение давления:

$$\nabla \cdot \left(\frac{1}{a_p}\nabla p\right) = \nabla \cdot \left(\frac{H(\vec{U})}{a_p}\right) = \sum_f \vec{S}\left(\frac{H(\vec{U})}{a_p}\right)_f$$

Алгоритм Simple

Simple позволяет связать уравнения Навье-Стокса с итерационной процедурой, которая может быть суммирована следующим образом:

1. Ставим граничные условия.
2. Решаем дискретизированное уравнение импульса для вычисления поля промежуточной скорости.
3. Вычисляем потоки массы на гранях ячеек.
4. Решаем уравнение давления и примените релаксацию.

5. Скорректируем потоки массы на гранях ячейки.
6. Скорректируем скорости на основе нового поля давления.
7. Обновим граничные условия.
8. Повторим до сближения.

Шаги 4 и 5 могут быть повторены в течение заданного количества времени для исправления неортогональности.

Алгоритм Piso

PISO является эффективным методом решения уравнений Навье-Стокса в нестационарных задачах. Основные отличия от алгоритма Simple заключаются в следующем:

1) Обычно дает более стабильные результаты и требует меньше процессорного времени, но подходит не для всех процессов.
2) Подходящие численные схемы для решения связанного уравнения скорости и давления.
3) Для ламинарного обратного шага шаг PISO быстрее, чем SIMPLE, но он медленнее в отношении потока через нагретое ребро.
4) Если импульс и скалярное уравнение имеют слабую связь или отсутствие связи, тогда PISO лучше, чем SIMPLEC

Алгоритм можно суммировать следующим образом:

1. Ставим граничные условия.
2. Решаем дискретизированное уравнение импульса для вычисления промежуточного поля скоростей.
3. Вычисляем потоки массы на гранях ячеек.
4. Решаем уравнение давления.
5. Скорректируем потоки массы на гранях ячеек.
6. Скорректируем скорости на основе нового поля давления.
7. Обновим граничные условия.
8. Повторим от 3 до предписанного количества раз.
9. Увеличиваем временной шаг и повторяем с 1.

Как уже было показано для алгоритма Simple, шаги 4 и 5 могут повторяться в течение заданного количества времени для исправления неортогональности

3. Практическая часть

Геометрия показана на рисунке 1 в которой все границы квадратной каверны - стены. Верхняя стена движется в x-направление со скоростью 1 м/с, в то время как остальные 3 неподвижны. Первоначально течение будет предполагаться ламинарным и будет решаться на однородной сетке с использованием icoFoam решатель для ламинарного, изотермического, несжимаемого течения. В ходе урока будет исследован эффект увеличения разрешения сетки и градуировки сетки по отношению к стенам. Наконец, число Рейнольдса потока будут увеличены и решатель pisoFoam будет использоваться для турбулентных, изотермический, несжимаемый поток.

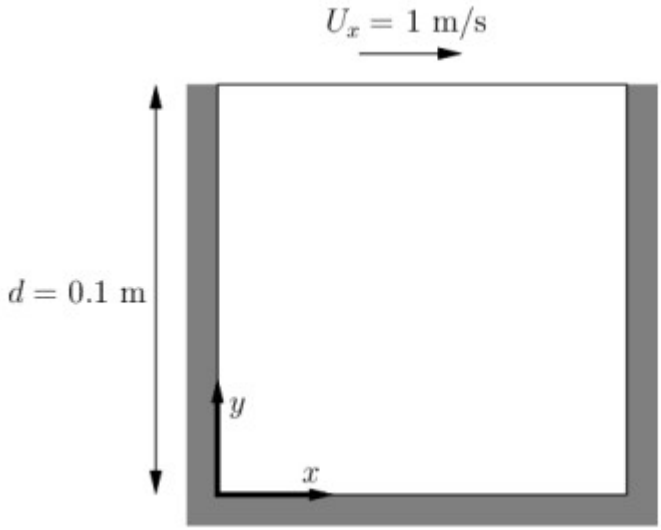

Рисунок 1: Геометрия приводной полости крышки.

Предварительная обработка

В процессе подготовки редактирования материалов дела и запуска первого в этом случае пользователь должен перейти в каталог case

```
cd $FOAM_RUN/tutorials/incompressible/icoFoam/cavity/cavity
```

Генерация сетки

OpenFOAM всегда работает в 3-мерной декартовой системе система координат и все геометрии генерируются в 3-х измерениях. OpenFOAM решает дело в 3 измерениях по умолчанию, но может быть проинструктирован решить его в 2 измерениях, указав "специальный" пустой граничное условие на границах, нормальных к (3-му) измерению, для которого не требуется никакого решения. Здесь

сетка должна быть толщиной в 1 клеточный слой, а пустой патчи плоские.

То полость домен состоит из квадрата длины стороны $d = 0.1\ \mathrm{m}$ в xy-самолет. Первоначально будет использоваться однородная сетка размером 20 на 20 ячеек. Блочная структура показана на рисунке

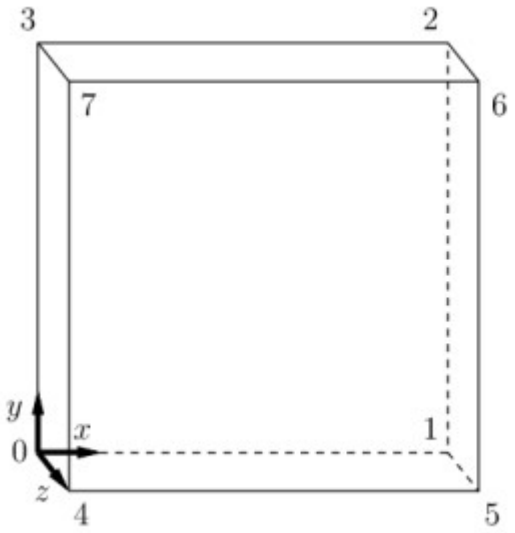

Рисунок 2: Блочная структура сетки для полости.

Граничные и начальные условия

Как только генерация сетки будет завершена, пользователь может посмотреть на эти начальные поля, настроенные для данного случая. Дело настроено так, чтобы начать его вовремя $t = 0$ s, поэтому исходные полевые данные хранятся
в 0 подкаталог полость каталог. То 0 подкаталог содержит 2 файла, p и U, по одному на каждое давление (p) и скорость ($\mathbf{U}$) поля, начальные значения и граничные условия которых должны быть заданы.

В файлах полевых данных есть 3 основные записи:

Размеры:	указывает размеры поля, здесь кинематический давление, т.е. $\mathrm{m}^2\mathrm{s}^{-2}$
Внутреннее поле:	данные внутреннего поля, которые могут быть однородными, описываемый одним значением; или неоднородный, где должны быть указаны все значения поля
Граничное поле:	данные граничного поля, включающие граничные условия и данные для всех граничных участков

Запуск приложения

Как и любой исполняемый файл UNIX/Linux, приложения OpenFOAM могут запускаться двумя способами: как процесс переднего плана, то есть тот, в котором оболочка ждет завершения команды, прежде чем дать командную строку; как фоновый процесс, который не должен быть завершен до того, как оболочка примет дополнительные команды.

По этому случаю мы будем запускать icoFoam на переднем плане. Решатель icoFoam выполняется либо путем ввода каталога case и ввода текста

```
icoFoam
```

Ход выполнения задания записывается в окно терминала. Он сообщает пользователю текущее время, максимальное число Куранта, начальные и конечные остатки для всех полей.

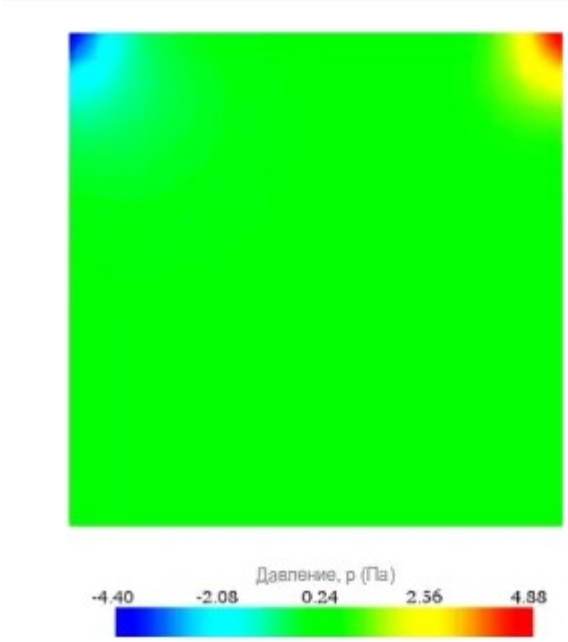

Рис. 3: Давление в корпусе полости.

Постобработка

Как только результаты записываются в каталоги времени, их можно просмотреть с помощью paraFoam.

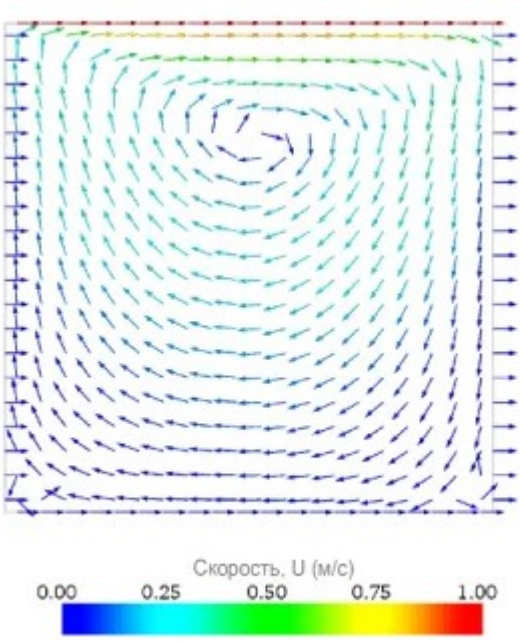

Рис. 4: Скорости в случае резонатора.

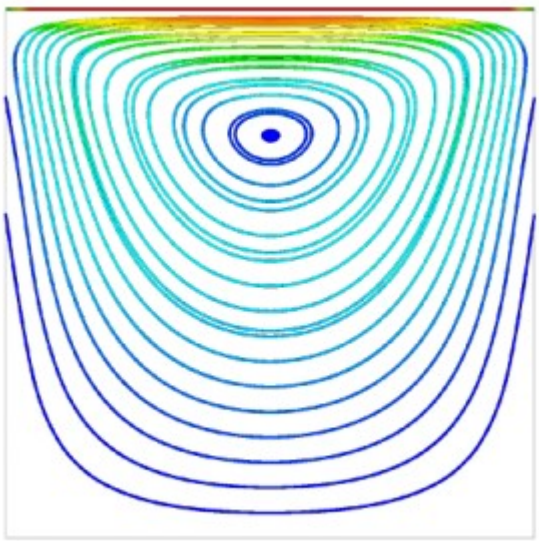

Рис. 5: Линии тока в корпусе резонатора.

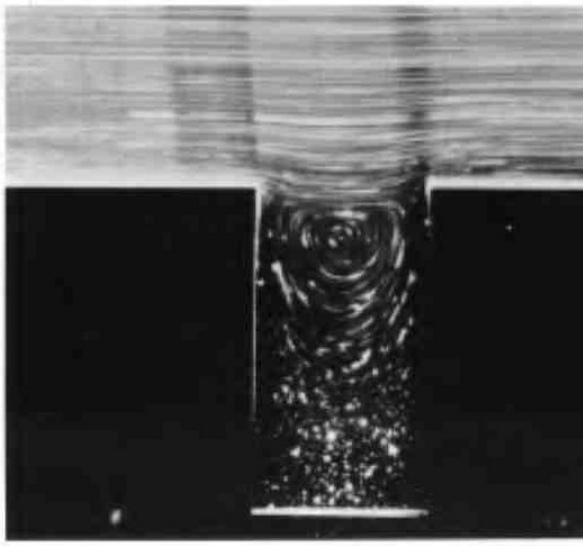

Рис. 6: Пример ползучего течения при обтекании прямоугольной каверны

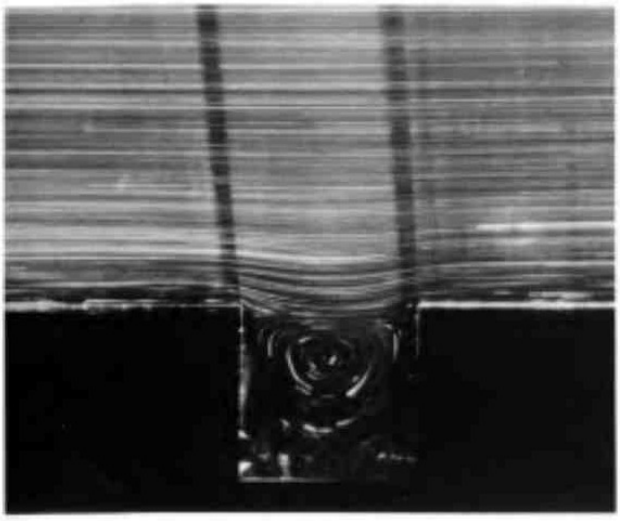

Рис. 7: Пример ползучего течения при обтекании прямоугольной каверны

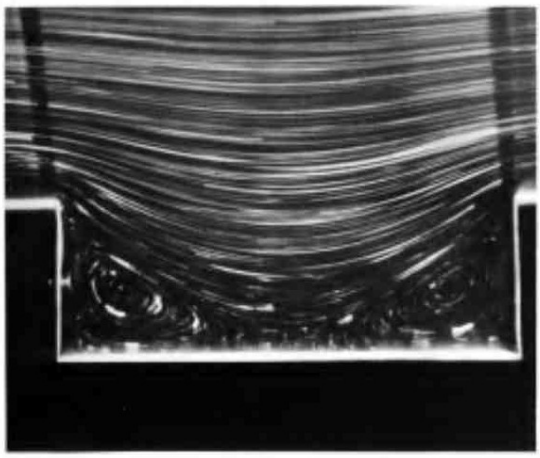

Рис. 8: Пример ползучего течения при обтекании прямоугольной каверны

ЗАКЛЮЧЕНИЕ

В ходе курсовой работы была изучена основа работы в OpenFOAM. Была освоена гексагональная сетка и утилита Blockmesh, с помощью которых была выявлена модель расчета давления-скорости. На практике было изучено численное решение задании о Каверне и выявлено совпадение с вычислениями.

СПИСОК ЛИТЕРАТУРЫ

1. OpenFOAM and STAR-CD. Integration, Interoperability and Symbiosis, Dr. Mark Olessen, EMCON Technologies// DANSIS-2007: New Trends In CFD, October 2007.
2. Интернет-ресурс http://www.openfoam.com/about/.
3. Интернет-ресурс https://www.openfoam.com/documentation/tutorial-guide/2-incompressible-flow/2.1-lid-driven-cavity-flow.
4. Васильев В.А., Ницкий А.Ю. Практика решений задач вычислительной гидродинамики тонких турбулентных слоев в щелевых уплотнениях питательных насосов на суперкомпьютерах и в распределенных вычислительных средах // Параллельные вычислительные технологии (ПаВТ'2009): Труды международной научной конференции (Нижний Новгород, 30 марта - 3 апреля 2009 г.). Челябинск: Изд-во ЮУрГУ, 2009. -С. 72-81
5. Patankar. S.V.,Spolding D.B. Int. J. Heat Mass Transfer, 15, 1972, p. 1787-1806
6. Патанкар С. Численные методы решения задач теплообмена и динамики жидкости // Москва: Энергоатомиздат, 150 с.
7. Альбом течений жидкости и газа: А56 Пер. с англ./Сост. М. Ван-Дайк.-М.: Мир, 1986.-184 с.

ПРИЛОЖЕНИЕ А

(обязательное)

Листинг программы

Алгоритм Simple

- Сохраним давление, рассчитанное на предыдущей итерации, так как требуется применить недо-релаксацию

```
p.storePrevIter();
```

- Определим уравнение для U

```
tmp<fvVectorMatrix> UEqn
(
fvm::div(phi, U) - fvm::laplacian(nu, U)
);

UEqn.relax();
```

- Решим предсказатель импульса

```
решить (UEqn == -fvc::grad(p));
```

- Обновим граничные условия для p

```
p.boundaryField().updateCoeffs();
```

- Вычислим a_pкоэффициент и U

```
volScalarField AU = UEqn().A();
U = UEqn().H()/AU;
UEqn.clear();
```

- Рассчитаем поток

```

```

```
phi = fvc::интерполяция(U) и сетка.Sf();
adjustPhi(phi, U, p);
```

- Определим и решим уравнение давления и повторите его для заданного числа шагов неортогонального корректора

```
fvScalarMatrix pEqn
(
fvm::laplacian(1.0/AU, p) == fvc::div(phi)
);
pEqn.setReference(pRefCell, pRefValue);
pEqn.solve();
```

- Исправляем поток

```
phi -= pEqn.flux();
```

- Вычисляем ошибки непрерывности

```
# включить "continuityErrs.H"
```

- Ослабим давление для корректора импульса и применим коррекцию

```
p.relax();
U -= fvc::grad(p)/AU;
U.correctBoundaryConditions();
```

- Проверяем сходимость и повторяем с самого начала, пока не будут выполнены критерии сходимости.

Значение начального остатка может быть получено при решении соответствующего уравнения с помощью метода initialResidual (). Возможны два синтаксиса:

```
eqnResidual = решить
(
UEqn() == -fvc::grad(p)
            ).initialResidual();
```

или, что то же самое, для уравнения давления, поскольку оно уже определено,

```
eqnResidual = pEqn.solve().initialResidual();
```

Алгоритм Piso

- Определяем уравнение для U

```
fvVectorMatrix UEqn
(
fvm::ддт(U)
+ fvm::div(phi, U)
- fvm::лапласиан(nu, U)
);
```

- Решаем предсказатель импульса

```
solve (UEqn == -fvc::grad(p));
```

- Вычисляем a_Pкоэффициент и U

```
volScalarField rUA = 1.0/UEqn().A();
U = rUA*UEqn().H();
```

- Вычисляем поток

```
phi = (fvc::interpolate(U) & mesh.Sf())
+ fvc::ДДТ(RUA, U, phi);
adjustPhi(phi, U, p);
```

- Определяем и решаем уравнение давления и повторите его для заданного числа шагов неортогонального корректора

```
fvScalarMatrix pEqn
(
fvm::лапласиан(rUA, p) == fvc::div(phi)
```

```
);
pEqn.setReference(pRefCell, pRefValue);
pEqn.solve();
```

- Исправляем поток

```
if (nonOrth == nNonOrthCorr)
{
phi -= pEqn.flux();
}
```

- Идет вычисление ошибок непрерывности

```
# включить "continuityErrs.H"
```

- Выполняем шаг корректора импульса

```
U -= rUA*fvc::grad(p);
U.correctBoundaryConditions();
```

- Повторяем расчет от a_Pначала до конца для предписанного числа шагов PISO-корректора.

ПЛАН-ГРАФИК

выполнения курсовой работы

обучающегося Шамаева И.Р.

Наименование этапа работ	Трудоемкость выполнения, час.	Процент к общей трудоемкости выполнения	Срок предъявления консультанту
Получение и согласование задания	0,3	0,8	29 неделя
Знакомство с литературой по теме курсовой работы	2,7	7,5	30 неделя
Знакомство с открытым численным пакетом OpenFOAM	10	30	32 неделя
Изучение алгоритмов	5	15,4	33 неделя
Изучение задания о каверне	5	13,8	34 неделя
Построение графической модели по изученным алгоритмам	10	24,2	37 неделя
Составление и оформление пояснительной записки и подготовка к защите	2,7	7,5	38 неделя
Защита	0,3	0,8	39 неделя
Итого	36	100	-