Курс «Формальная спецификация и верификация», 2011-2012 уч.год

Освоение курса предполагает посещение лекций, выполнение ряда заданий, сдачу зачета и экзамена. За каждый этап студент получает некоторое количество баллов, сумма которых определяет оценку за курс.

1 Работа в семестре

В рамках семестра студенты должны посещать лекции и семинарские занятия, выполнять различные практические задания. Формы работы и баллы за них в это время:

1. Написание контрольной работы №1. Дает до 10 баллов. Дата проведения назначается семинаристом.
2. Написание контрольной работы №2. Дает до 10 баллов. Дата проведения назначается семинаристом.
3. Выполнение практического задания по RAISE методу. Дает до 20 баллов. Является обязательным заданием для получения зачета по курсу. Дата сдачи задания назначается семинаристом.
4. Выполнение практического задания по Dafny. Дает до 20 баллов. Является опциональным заданием. Дата сдачи выполненного задания - до начала декабря 2011 года.
5. Выполнение практического задания по PVS. Дает до 20 баллов. Является опциональным заданием. Дата сдачи выполненного задания – до начала декабря 2011 года.
6. Участие в семинарских занятиях. Может быть до 10 баллов.
7. Участие в лекциях. Может дать до 5 баллов на каждой лекции по усмотрению лектора.

2 Контрольные работы

В семестре предполагается проведение двух контрольных работ. Первая контрольная работа посвящена языку RSL. Вторая контрольная работа посвящена методам Флойда и инструменту PVS. Контрольная работа предполагает выполнение задач, возможных на экзамене, за ограниченное время (40 минут на одну контрольную работу). Каждая контрольная работа состоит из двух задач. За одну контрольную работу можно получить до 10 баллов. Контрольные работы проводятся в рамках семинарских занятий. Условия проведения контрольных работ идентичны экзаменационным условиям: можно пользоваться любой печатной и рукописной литературой, но запрещается пользоваться любыми электронными средствами (компьютерами, телефонами и т.п.), а также запрещается общение со всеми, кроме экзаменаторов и семинаристов.

3 Практические задания

Для закрепления полученных знаний и их практической апробации студентам предлагается выполнить ряд практических заданий. Есть практические задания трех видов:

- задания по языку RSL и RAISE методу;

- задания по спецификации и верификации программ при помощи инструмента Dafny;
- задание по спецификации программ при помощи инструмента Frama-C и языка ACSL и их верификации при помощи инструмента PVS.

Практическое задание по языку RSL и RAISE методу является **обязательным** для получения зачета по курсу. Его содержание приведено ниже.

За каждое задание (в том числе и за задание по RAISE) студент получает некоторое количество баллов, соответствующих качеству выполнения задания. Чем лучше выполнено задание, тем больше баллов за него получит студент. Принцип выставления балла за выполненное задание по RAISE определяет семинарист.

Практическое задание по Dafny предполагает написание спецификации некоторого набора алгоритмов и их (автоматической) верификации средствами инструмента Dafny. Пример практического задания по Dafny приводится ниже. Аналогично, с заданием по Frama-C + PVS. Чтобы получить задание по Dafny, надо написать письмо Евгению Корныхину (kornevgen@cmc.msu.ru). Чтобы получить задание по Frama-C + PVS надо написать письмо Алексею Хорошилову (khoroshilov@ispras.ru). Задание по этим инструментам можно взять не только лично, но и на группу студентов (это надо указать в письме). В этом случае количество баллов, которое получит каждый студент в группе, равно количеству баллов за выполненное задание, поделенное на количество студентов в группе. В ответном письме на письмо с запросом задания (кроме самого задания) будет написан его срок выполнения.

4 Зачет по курсу

Для получения зачета по курсу необходимо:

1. Успешно написать письменную работу.
2. Успешно сдать семинаристу выполненное практическое задание по языку RSL и RAISE методу.

Письменная работа включает в себя одну задачу (на методы Флойда). В этой задаче будет написан алгоритм в виде блок-схемы и пред- и пост- условие. Необходимо при помощи методов Флойда доказать частичную и полную корректность этого алгоритма относительно данных пред- и пост-условия.

Условия выполнения этой письменной работы идентичны экзаменационным условиям: можно пользоваться любой печатной и рукописной литературой, но запрещается пользоваться любыми электронными средствами (компьютерами, телефонами и т.п.), а также запрещается общение со всеми, кроме экзаменаторов и семинаристов.

Письменная работа проводится одновременно во всех группах.

Выполнение письменной работы не дает дополнительных баллов по курсу.

Возможно, что до зачетной сессии будет устроена предварительная попытка написания этой письменной работы. Если студент успешно справляется с нею, письменную работу на зачете ему писать не нужно.

5 Экзамен по курсу

Экзамен по курсу письменный. Он предполагает выполнение ряда (8-10) небольших заданий по темам, рассказанным на лекциях и отработанных на семинарских занятиях. Типы и примеры заданий будут

опубликованы перед экзаменом на странице курса (http://sp.cmc.msu.ru/courses/fmsp). На экзамене можно пользоваться любой печатной и рукописной литературой, но запрещается пользоваться любыми электронными средствами (компьютерами, телефонами и т.п.), а также запрещается общение со всеми, кроме экзаменаторов и семинаристов.

Студент может не писать экзамен по курсу, если количество набранных к этому моменту баллов достаточно для получения желаемой им оценки по курсу.

Длительность экзамена – 120 минут. Экзамен проводится одновременно у всех групп.

6 Оценка по курсу

В зависимости от суммы набранных студентом баллов будет выставлена оценка по курсу.

Если сумма баллов – от 30 до 59 баллов – студент получает оценку «удовлетворительно».

Если сумма баллов – от 60 до 79 баллов – студент получает оценку «хорошо».

Если сумма баллов – от 80 и выше – студент получает оценку «отлично».

7 Самое важное в одном абзаце

Этот раздел – краткая выжимка основной информации из этого документа.

В рамках семестра надо набирать баллы: работая на семинарах, на лекциях, выполняя практические задания, написав экзамен. В зависимости от суммы баллов будет поставлена оценка (см. пункт 6). Есть практическое задание по RAISE, оно обязательное для зачета, дает до 20 баллов. Есть задание по Dafny, оно необязательное для зачета, дает до 20 баллов. Есть задание по PVS, оно необязательное для зачета, дает до 20 баллов. На семинарах проводятся две контрольные. За каждую контрольную можно получить до 10 баллов. Контрольная – это частичная «репетиция» экзамена (задачи те же, условия те же, времени в два раза больше). Семинарист может поставить еще до 10 баллов за работу на семинарах. На лекциях даются небольшие задания, за которые тоже можно получить баллы. Для получения зачета нужно обязательно решить задачу на Флойда, вне зависимости от того, сколько получено баллов, и сдать семинаристу задание по RAISE. На экзамене будет много небольших задачек, их типы и примеры лекторы опубликуют перед экзаменом на странице курса. На экзамене можно пользоваться конспектами и книжками, но нельзя пользоваться компьютерами и нельзя разговаривать.

Приложение 1. Практическое задание по языку RSL и RAISE методу

Примечание: чтобы получить и сдать задание по RAISE методу, студент обращается к своему семинаристу.

Задание: вариант задания – это, зачастую, некий управляющий модуль некой информационной системы с некоторым интерфейсом (список систем есть в методичке Петренко-Кузьменкова 2008 года). Требуется расширить функциональность этого модуля, предложив не менее **двух** новых качественно различных возможностей. Согласовать эти возможности с семинаристом. Затем надо написать три спецификации (и подготовить ответ на ряд вопросов к ним, см. ниже).

Первая спецификация – это алгебраическая спецификация. В ней надо сформулировать обозримое поведение системы (т.е. поведение системы с точки зрения её пользователя). Спецификация должна быть настолько полной, насколько позволяет постановка задания, но не более.

Примечание: для описания обозримого поведения можно использовать и модельные спецификации, но поскольку разницу между такими модельными спецификациями и модельными спецификациями, описывающими реализацию системы, понять непросто, использовать модельные спецификации для описания обозримого поведения системы в рамках практического задания **запрещается**.

Вторая спецификация – это неявная спецификация. В ней по сравнению с первой спецификацией нужно сделать выбор реализационных структур данных для типов (эти же структуры данных без изменений будут в третьей, явной, спецификации). Для всех операций системы (не включая вспомогательные, если таковые появились) надо описать пред- и пост- условия на основе выбранных структур данных, но не делая при этом никаких предположений об алгоритмах выполнения операций. Т.е. надо описать, что делает операция, какую она решает задачу, но не описывать то, как она это делает, каким алгоритмом (например, для задачи проверки вхождения подстроку в строку существует множество различных алгоритмов, Бойера-Мура, Морриса-Пратта, Кнута-Морриса-Пратта, но все они решают одну и ту же задачу).

Одной из часто встречающихся ошибок при написании данной спецификации является желание выразить в виде задачи некоторое вполне очевидное решение. При этом игнорируются все не очевидные автору в момент написания спецификации другие реализации. Например, если пусть нужно описать журнал записей и операцию добавления в журнал одной записи, причем для представления журнала используется список. Тогда необоснованной будет неявная спецификация, в которой запись дописывается всегда в конец списка. Вместо этого, постусловие операции добавления должно содержать условие, что запись добавлена в список (не обязательно в конец!), условие о том, что ничего из списка не исчезло, условие о том, что ничего кроме этого не было добавлено.

Итого, в плюс к неявной спецификации необходимо подготовить (устно) обоснование сделанного выбора структур данных.

Для основных типов кроме функций с пред- и пост- условиями надо описать инварианты (описать в виде аксиом условия, выделяющие согласованные значения структур данных, представляющих эти типы).

Третья спецификация – это явная спецификация, в которой надо сделать проектные решения относительно алгоритмов выполнения операций, не меняя структур данных из второй спецификации, так, чтобы добиться баланса эффективности выполнения операций.

Кроме текста спецификации на RSL надо подготовить (устно) обоснование сделанного выбора именно таких алгоритмов по каждой основной операции. Например, «вставляю ближе к началу, потому что есть функция поиска, которая ищет данные с начала списка; значит, вставка в начало позволит быстрее сделать поиск».

В спецификациях удобно использовать множества. Поскольку у нас задания не такие сложные и не предполагают построения сложных структур данных, то для оценки эффективности их реализаций становится важно, как реализованы множества (на массивах, битовых векторах, деревьях т.п.). Грубо говоря, в таких задачах множество – это тоже абстрактная структура данных. Однако RSL не фиксирует какой-либо одной реализации множеств. Тем самым, в наших заданиях нельзя серьезно говорить об эффективности реализации, если за основу структур данных для типов выбраны множества. Поэтому в явных/неявных спецификациях **нельзя пользоваться множествами** как реализационными структурами данных. Вместо этого надо пользоваться списками, структурами (в смысле, struct языка C или record языка Паскаль), перечислениями, записав на них инварианты.

Кроме того все спецификации должны быть понятными (в основном, это касается формулировки требований в неявных спецификациях). Т.е., например, если требование логично разбивается на несколько независимых частей, каждая из которых представляется сложным выражением, нужно оформить каждое такое выражение в виде отдельной функции с понятным названием.

Итак, требования к результатам выполнения практического задания по языку RSL и RAISE методу (если какие-нибудь из этих пунктов не выполнены, преподаватель снижает балл за решение) :

1. Алгебраическая спецификация должна описывать только видимое поведение системы.
2. Структуры данных в явной и неявной спецификации должны совпадать.
3. Прислано объяснение выбора структур данных и алгоритмов.
4. Выбор структур данных и алгоритмов обоснован (на 5 курсе студенты знают уже достаточно алгоритмов и структур данных и способов обоснования их выбора), эффективность выполнения операций должна быть главной идеей обоснования.
5. Явная спецификация должна соответствовать тем алгоритмам, которые заявлены в объяснении.
6. Явная спецификация должна полностью реализовывать все требования, записанные в неявной спецификации.
7. Неявная спецификация не должна включать каких-либо проектных решений относительно алгоритмов реализации операций.
8. Неявная спецификация должна содержать инварианты основных типов. Инварианты должны быть оформлены в виде аксиом.
9. Неявные и явные спецификации не могут использовать множества в качестве реализационных структур данных (но могут использоваться в качестве модельных структур данных, например, в какой-нибудь вспомогательной функции в неявной спецификации).
10. Все спецификации должны быть синтаксически и семантически корректны.
11. Все спецификации должны быть понятны.

Приложение 2. Пример практического задания по Dafny

Примечание: чтобы получить и сдать задание по Dafny, студент посылает письмо Евгению Корныхину kornevgen@cmc.msu.ru до 20 ноября 2011 года. Срок сдачи – до 10 декабря 2011 года.

Задание: написать спецификацию и верифицировать при помощи инструмента Dafny данный алгоритм.

Пример: алгоритм проверки связности графа.

```
method by_edge( g : seq< set<int> >, start : set<int> ) returns ( finish : set<int> )
{
    var i := 0; finish := { };
    while ( i < |g| )    {
        if ( i in start ) { finish := finish + g[i]; }
        i := i + 1;
    }
}

method connected( g : seq< set<int> >, start : int, finish : int ) returns ( cond : bool )
{
```

```
    var todo, reached := {start}, { };
    while ( todo != { } && finish !in todo ) {
        reached := reached + todo;
        var vs := by_edge(g, todo);
        todo := vs - reached;
    }
    return finish in todo;
}
```

Приложение 3. Пример практического задания по Frama-C + PVS

Примечание: чтобы получить и сдать задание по Frama-C + PVS, студент посылает письмо Алексею Хорошилову khoroshilov@ispras.ru до 20 ноября 2011 года. Срок сдачи – до 10 декабря 2011 года.

Задание: средствами инструмента PVS доказать лемму.

Пример:

```
intersect_why: THEORY
BEGIN
  is_intset_ex: LEMMA
    (FORALL (int_P_int_M_s_15_11_at_L: memory[int_P, int32]):
    (FORALL (s_15: pointer[int_P]):
    (FORALL (n_3: int):
    (is_intset(s_15, n_3, int_P_int_M_s_15_11_at_L) IFF
    (FORALL (i_7: int):
    (FORALL (j_1: int):
    (0 <= i_7 AND (i_7 < n_3 AND (0 <= j_1 AND (j_1 < n_3 AND
    i_7 /= j_1))) IMPLIES
    integer_of_int32(select[int32,
                    int_P](int_P_int_M_s_15_11_at_L, shift[int_P](s_15, i_7))) /=
    integer_of_int32(select[int32,
                    int_P](int_P_int_M_s_15_11_at_L, shift[int_P](s_15,
j_1)))))))))))

END intersect_why
```

Приложение 4. Некоторые типы экзаменационных задач по RSL и RAISE методу

1. По данной явной спецификации построить эквивалентную неявную спецификацию без использования рекурсии и циклов.

Пример:

```
variable x : Int, y : Int
value f : Unit -~-> write x, y Nat
   f() is (if x + y > 0 then y := x end; if y > 1 then y := 0 end; x + y)
```

Ответ:

```
variable x : Int, y : Int
value f : Unit -~-> write x, y Nat
f() as n
post x` = x /\ n = x + y /\  ( if x` + y` = 0 then
```

```
            if x` > 1 then y = 0 else y = x` end
        else
            if y` > 1 then y = 0 else y = y` end      end )
```

2. По данной алгебраической спецификации вычислить значение выражения, если это возможно сделать. В ответе представить процесс вычисления в виде переписывания термов.
Пример: выражение: first(add(add(add(add(add(empty, 5), 100), 0), 2), 20)).
алгебраическая спецификация:

```
scheme A0 = class
   type S
   value empty : S,
      add : S >< Nat -> S,
      first : S -~-> Nat
   axiom all s : S, e : Nat :-
      first(add(s, e)) is if e > 10 then 2 * first(s)
                       elsif e > 5 then first(s)
                       else 0 end
end
```

Ответ: first(add(add(add(add(add(empty, 5), 100), 0), 2), 20)) = 2 * first(add(add(add(add(empty, 5), 100), 0), 2)) = 2 * 0 = 0. Вычисление возможно.

3. Указать, какие из следующих стилевых характеристик спецификации (явная, неявная, алгебраическая, модельная, аппликативная, императивная, объектная, формальная, неформальная) относятся к данной спецификации.
Пример: (в качестве спецификации может быть любой текст, который встретился в курса, а именно RSL-спецификация, Dafny-программа, PVS-теоремы)

```
scheme S = class
   type T, E
   variable x : Nat
end
```

Ответ: императивная, формальная.

4. Дать определение термина.
Пример: предусловие операции.
Ответ: Условие на входные данные операции, состояние исполняющего операцию объекта, и состояние среды, в которой находится объект, которое должен обеспечить тот, кто вызвал операцию.

5. Привести **одно** отличие двух данных понятий.
Пример: VDM и RAISE.
Ответ: в RAISE нельзя менять выбранное уточнение типов. в VDM можно.

6. Укажите в приведенном тексте **одно неполное** требование.
Пример: Реализовать систему по сбору статистики посещения сайта. При этом должна быть реализована панель администратора, где он мог бы посмотреть статистику посещений, составить отчеты, указав временные интервалы, разделы сайта и т.п.
Ответ: не указано, имеет ли доступ к статистике кто-либо, отличный от администратора.

7. Запишите на RSL инвариант типа S согласно заданию, не меняя определений типов. При необходимости можете добавить новые разделы спецификации.
Пример: type E, S = E-list
S является стеком ограниченного размера
Ответ:
value size : Nat
type E, S = E-list
axiom all s : S :- len s <= size

8. Чему равно значение каждого из следующих выражений? Если ответ зависит от какие-либо переменных, максимально упростить выражение (типы всех переменных таковы, что все написанные выражения семантически корректны).
Пример:
а) <. x**2 | x in <.1 .. 3.> :- abs x < 10 .>
б) {x, y, x} \ { x }
в) [1 +> 2] !! [1 +> 3] union [2 +> 2]
г) x > 0 /\ x + y < 1 /\ y > 2
д) x isin s => x isin (s union t)
Ответ: а) <.1, 4, 9 .>, б) {y}\{x}, в) [1 +> 3, 2 +> 2], г) false, д) true

9. Формализовать на RSL в виде аксиомы следующую фразу русского языка, правильно отразив ее смысл.
Пример : “Если вчера Петров прогулял два занятия, то сегодня только одно.”
Ответ:
type человек, день
value Петров : человек, вчера : день, сегодня : день
value прогулял: человек >< день -> Nat
axiom прогулял(Петров, вчера) = 2 /\ прогулял(Петров, сегодня) = 1

10. Приведите пример инвариантного свойства, которое должна поддерживать данная подсистема/система.
Пример: диспетчер виртуальной памяти.
Ответ: нет двух записей про один виртуальный адрес, которым соответствуют разные физические адреса.

11. Для данной подсистемы/системы привести **один** пример операции, которую имеет смысл сначала определить частично (т.е. не тотально). Укажите условие, при котором эта операция будет не определена в такой спецификации.
Пример: диспетчер виртуальной памяти.
Ответ: операция получения виртуального адреса по физическому. Имеет смысл ее сначала определить частичной, например, не определять для виртуальных адресов, которым в данный момент не соответствует какой-либо физический адрес.

Приложение 5. Некоторые экзаменационные задания по методам Флойда

Будут опубликованы позднее