На решение одной задачи должно уходить 5-10 минут

RSL и RAISE метод

“практические” задания

A1. 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;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 y = x` else y = y` end )
```

A2. 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.

A3. Указать, какие из *стилевых* характеристик относятся к данной спецификации (явная, неявная, алгебраическая, модельная, аппликативная, императивная, объектная, формальная, неформальная).

Пример: (в качестве спецификации может быть любой текст, который встретился в курса, а именно RSL-спецификация, PVS-теоремы)

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

Ответ: императивная, формальная.

A4. Запишите на RSL инвариант типа S согласно заданию. При необходимости

можете добавить новые разделы спецификации, не изменяя определения типов, присутствующих в задании.
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

A5. Чему равно значение каждого из следующих выражений на RSL? Если ответ зависит от какие-либо переменных, записать эквивалентное выражение, максимально упростив исходное 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

A6. Формализовать на RSL в виде аксиомы следующую фразу русского языка, правильно отразив ее смысл. При необходимости можно ввести дополнительные value и type.
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
```

A7. Приведите пример инвариантного свойства состояния указанной подсистемы/системы или её объекта (объектов) управления, которое она должна поддерживать.
Пример: диспетчер виртуальной памяти.
Ответ: нет двух записей про один виртуальный адрес, которым соответствуют разные физические адреса.

A8. Для данной подсистемы/системы привести *один* пример операции, которую имеет смысл на ранних этапах спецификации 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.

“теоретические” задания

Б1. Дать определение термина в одном-двух предложениях. Определение должно пояснять смысл 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.

Б2. Привести *одно* отличие двух данных понятий.
Пример: VDM и RAISE.
Ответ: в RAISE нельзя менять выбранное уточнение типов. в VDM можно.

Методы Флойда

“практические” задания

А1. Для данных пред- (pre) и пост- (post) условия, записанных на RSL, предложите реализацию функции без побочных эффектов, которая является частично корректной относительно них. Для записи функции так же используйте RSL, но без кванторов и рекурсии. Все используемые операции RSL не должны в момент вычисления давать chaos. Функция должна завершаться на максимальном числе входных данных, удовлетворяющих предусловию.
Пример:
pre(x, y) = hd x > hd y
post(x, y, z) = z(2) isin elems tl tl x
Ответ:

```
value f : Int-list >< Int-list -~-> Int-list
f(x, y) is if len x <= 2 then while true do skip end;<..> else tl x
end
pre len x > 0 /\ len y > 0 /\ hd x > hd y
```

А2. В блок-схеме программы выделена подсхема, в которую входит ровно одна дуга и из которой выходит ровно одна дуга. Известно, что про эту схему доказана полная корректность относительно некоторых пред- и пост- условий. При этом дуге, ведущей в подсхему, была сопоставлена точка сечения A и оценочная функция uA, а дуге, исходящей из подсхемы, точка сечения B и оценочная функция uB. Фундированное множество обозначено как W. Для данных uA, uB, W предложить содержимое подсхемы (в виде программы на RSL) и утверждение в точке A, гарантирующее достижимость B из A на максимальном числе данных. x - входная переменная программы, y1, y2 - промежуточные переменные.
Пример:

uA = len y1 + len y2
uB = uA
W = (Nat, <)
Ответ:
if y2 != <..> then y2 := tl y2 else y1 := tl y1 end;
утверждение в точке A: y1 ^ y2 != <..>

A3. Какое требуется минимальное число точек сечения, чтобы доказать частичную корректность *хотя бы одной* программы, имеющей указанные особенности. Ответ кратко обосновать.
Пример:
блок-схема содержит 2 вложенных цикла
Ответ: 1: точка сечения находится перед входом во внутренний цикл. Нуля недостаточно, т.к. внутренний цикл не будет иметь точку сечения.

A4. Что из перечисленного может быть индуктивным утверждением?(все указанные переменные целочисленные, эквивалентные преобразования формул не производились, за исключением, быть может, опускания несущественных скобок)
Пример:
а) t > 0
б) t
в) all t : Int :- (t > 0 => t+1 > 0)
г) (Nat union { -1, -2 }, <=)
д) all y : Int :- exists t : Int :- (y > t => t > 0)
Ответ: а, в, д

A5. Что из перечисленного может быть оценочной функцией ? (все указанные переменные целочисленные, эквивалентные преобразования формул не производились, за исключением, быть может, опускания несущественных скобок)
Пример:
а) t > 0
б) t
в) all t : Int :- (t > 0 => t+1 > 0)
г) (Nat union { -1, -2 }, <=)
д) all y : Int :- exists t : Int :- (y > t => t > 0)
Ответ: а, б, в, д

A6. Что из перечисленного может быть фундированным множеством ? (все указанные переменные целочисленные, эквивалентные преобразования формул не производились, за исключением, быть может, опускания несущественных скобок)
Пример:
а) t > 0
б) t
в) all t : Int :- (t > 0 => t+1 > 0)

г) (Nat union { -1, -2 }, <=)
д) all y : Int :- exists t : Int :- (y > t => t > 0)
Ответ: г

A7. Что из перечисленного может быть условием корректности ? (все указанные переменные целочисленные, эквивалентные преобразования формул не производились, за исключением, быть может, опускания несущественных скобок)
Пример:
а) t > 0
б) t
в) all t : Int :- (t > 0 => t+1 > 0)
г) (Nat union { -1, -2 }, <=)
д) all y : Int :- exists t : Int :- (y > t => t > 0)
Ответ: в

A8. Что из перечисленного может быть условием верификации ? (все указанные переменные целочисленные, эквивалентные преобразования формул не производились, за исключением, быть может, опускания несущественных скобок)
Пример:
а) t > 0
б) t
в) all t : Int :- (t > 0 /\ true => t+1 > 0)
г) (Nat union { -1, -2 }, <=)
д) all y : Int :- exists t : Int :- (y > t => t > 0)
Ответ: в

A9. Что из перечисленного может быть условием завершимости ? (все указанные переменные целочисленные, эквивалентные преобразования формул не производились, за исключением, быть может, опускания несущественных скобок)
Пример:
а) t > 0
б) t
в) all t : Int :- (t > 0 => t+1 > 0)
г) (Nat union { -1, -2 }, <=)
д) all y : Int :- exists t : Int :- (y > t => t > 0)
Ответ: ничего

A10. Доказать или опровергнуть теорему о частичной или полной корректности. Заглавными буквами обозначены программы, строчными - формулы.
Пример:
Верно ли, что для любых a, b если {a}P{b}, то {b}P{a} ?
Ответ: неверно; контрпример: a = false, b любое, P не зацикливается хотя бы на данных, для которых выполнено b. {false}P{b} = true, {b}P{false} = false.

A11. Дано выражение. Может ли оно быть оценочной функцией и, если может, то при каких условиях? Свой ответ кратко аргументируйте по определению.
Пример: выражение: x (x - очередь из целых чисел)

Ответ: да при следующих условиях: фундированное множество - множество очередей из целых чисел, отношение частичного порядка - x <= y <=> длина x <= длина y. Множество будет фундированным, т.к. любая очередь >= пустой очереди, чья длина равна 0.

“теоретические” задания

Б1. Дать определение термина в одном-двух предложениях. Определение должно пояснять смысл термина.

Пример: частичная корректность.

Ответ: это отношение на программах и парах формул: {a}P{b} т.и т.т.,к. на всех входных данных, удовлетворяющих a, если P на них завершается, то для них и результата программы выполнено b.

Б2. Приведите *одно отличие и одно сходство* данных двух понятий.

Пример: полная корректность и частичная корректность.

Ответ: отличие - первое всегда ложно на тех данных из предусловия, на которых программа зацикливается, а второе истинно; сходство - это всё отношения на программах и парах формул.

Инструмент PVS

“практические” задания

A1. Каково минимальное количество листовых вершин в дереве доказательства следующей теоремы (не используя assert и grind) ?

Пример: (A IMPLIES B AND C) AND A IMPLIES B

Ответ: 3, нужно два split’a.

A2. Приведите PVS теорему, в которой бы встречался указанный кусок доказательства (последовательность команд PVS, все команды завершаются результативно).

Пример: (skosimp) (assert)

Ответ: FORALL (x : int) : FORALL (y : int) : (x - y >= 0 AND x + y >= 0 IMPLIES x >= 0)

A3. Можно ли использовать PVS для приведенной цели? Если можно, напишите *одно* предложение, как.

Пример: формальная спецификация функциональных свойств программ.

Ответ: можно, теории PVS - и есть формальные спецификации функциональных свойств программ.

“теоретические” задания

Б1. Что делает указанная команда PVS ?

Пример: split

Ответ: разбивает конъюнкцию в консеквенте на несколько подцелей доказательства, каждая соответствует своему конъюнкту; аналогично с дизъюнкцией в антецеденте.

Б2. Напишите два сходства приведенных двух понятий.
Пример: антецедент и консеквент.
Ответ: 1) и то, и другое - формула. 2) оба вида формул появляются при доказательстве теорем.