При использовании листа обратной связи в столбце «Урок» в скобках указан условный «уровень квалификации» учащегося, необходимый для выполнения данного вида работы (то есть каков должен быть уровень освоения результата). Самооценка осуществляется в ходе выполнения заданий с одновременной фиксацией в листах обратной связи рекомендаций/вопросов учителя. Для своих комментариев учителю предлагается использовать специфические для данного предмета условные обозначения для обратной связи:

|  |  |  |
| --- | --- | --- |
| 1 | **?** | Рассмотрены не все варианты. |
| 2 |  | Необходимо ветвление задачи, например – рассмотрение ее частных случаев |
| 3 |  A  AB B | Необходимо разделить данный модуль на функциональные компоненты. |
| 4 |  A AB  B | Необходимо соединить несколько модулей в один. |

Основанием для анализа промежуточных результатов является наблюдение за работой учащихся за компьютером и их текущая самооценка.

Уровни самооценки обозначаются следующими знаками: ☺ - ясность на всех этапах выполнения задания, 😐 - ясность большинства компонентов вопроса, ☹- выполнено без понимания. Отсутствие символа самооценки после завершения этапа означает, что учащемуся непонятно, как выполнять задание, нужна подсказка учителя. Допускается изменение самооценки ранее выполненных компонентов по мере выполнения всего задания, как это указано в примере.

Уровень результата определяется следующим образом:

Уровень результата = (Сумма баллов по уровням I-V)\*(количество баллов по уровню VI).

Помимо листа обратной связи учитель может предложить учащемуся представить какой-либо фрагмент работы, например, фрагмент программы письменно на отдельном листе бумаги и затем прокомментировать данный фрагмент, чтобы помочь учащемуся обнаружить возможные причины его затруднений.

**Лист обратной связи**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Тема:** | **Подпрограммы в Паскале** | **Фамилия, имя:** |  | **Класс:** |  |
| **Урок** | **Задание** | **Ожидаемый результат, значимость (вес)** | **Самооценка** | **Рекомендации** |
| 1, 2(I) | Выделите структурные элементы задания, указав письменно (на отдельном листе) назначение и обосновав необходимость каждого | 1) Выделение элементов: 0,4 (**анализ** задания)2)Описание назначения: 0,5 (**знание**, **понимание**)3) Обоснование: 0,1 (**оценка** значимости) |  |  |
| 3(II) | Представьте структурные элементы через известные алгоритмические конструкции (по возможности) | 1) Знание назначения алгоритмов: 1 (**знание**, **понимание**) |  |  |
| 4, 5(III) | Составьте подпрограммы, соответствующие выделенным структурным элементам | 1) **Знание** текстов основных алгоритмов: 0,52) Умение записать их как подпрограммы: 0,5 (**применение**) |  |  |
| 6(IV) | Выберите целесообразные способы обмена данными между структурными элементами программы, скорректируйте структурную схему | 1) Знание основных способов обмена данными между подпрограммами:0,2 (**знание**, **понимание**)2) Выбор лучшего способа обмена: 0,5 (**анализ**)3) Изменение схемы для оптимизации обмена данными между подпрограммами: 0,3 (**синтез**) |  |  |
| 7(V) | Осуществите сборку и отладку программы. | 1) Сборка программы: 0,7 (**синтез**)2) Отладка: 0,3 (**знание**, **применение**) |  |  |
| 8(VI) | Проведите защиту заданий | 1) **Знание** основных алгоритмов: 0,12) Место алгоритма в задаче: 0,2 (**понимание**)3) **Применение** знаний для выполнения заданий учителя (например: изменить способ масштабирования и пр.): 0,44) **Оценка** целесообразности внесения в программу изменений, предлагаемых преподавателем: 0,3 |  |  |

**Найти разность средних арифметических значений двух вещественных массивов из 10 элементов.**

Как видно из условия, для двух массивов требуется найти одну и ту же величину - среднее арифметическое. Следовательно, логичным будет оформить его нахождение в виде подпрограммы, которая сможет работать с разными массивами.

*program dif\_average;*

*const n = 10;*

*type mas = array[1 .. n] of real;*

*var a, b : mas;*

*i : integer;*

*dif, av\_a, av\_b : real;*

*procedure average(x : mas; var av : real); {1}*

*var i : integer;*

*begin*

*av := 0;*

*for i := 1 to n do*

*av := av + x[i];*

*av := av / n;*

*end; {2}*

*begin*

*for i := 1 to n do*

*read(a[i]);*

*for i := 1 to n do*

*read(b[i]);*

*average(a, av\_a); {3}*

*average(b, av\_b); {4}*

*dif := av\_a - av\_b;*

*writeln('Разность значений ', dif:6:2)*

*end.*

Описание процедуры average расположено в строках с {1} по {2}. В строках, помеченных цифрами {3} и {4}, эта процедура вызывается сначала для обработки массива а, затем - массива b. Эти массивы передаются а качестве аргументов. Результат вычисления среднего арифметического возвращается в главную программу через второй параметр процедуры.

**Функции**

Описание функции отличается от описания процедуры незначительно:

*function имя [(список параметров)] : тип; { заголовок }*

*разделы описаний*

*begin*

*раздел операторов*

*имя := выражение;*

*end;*

Функция вычисляет одно значение, которое передается через ее имя. Следовательно, в заголовке должен быть описан тип этого значения, а в теле функции - оператор, присваивающий вычисленное значение ее имени.

*program dif\_average1;*

*const n = 3;*

*type mas = array[1 .. n] of real;*

*var*

*a, b : mas;*

*i : integer;*

*dif : real;*

*function average(x : mas) : real; {1}*

*var i : integer; {2}*

*av : real;*

*begin*

*av := 0;*

*for i := 1 to n do av := av + x[i];*

*average := av / n; {3}*

*end;*

*begin*

*for i := 1 to n do read(a[i]);*

*for i := 1 to n do read(b[i]);*

*dif := average(a) - average(b); {4}*

*writeln('Разность значений ', dif:6:2)*

*end.*

Оператор, помеченный комментарием {1}, представляет собой заголовок функции. Тип функции определен как вещественный, потому что такой тип имеет среднее арифметическое элементов вещественного массива. Оператор {3} присваивает вычисленное значение имени функции. В операторе {4} функция вызывается дважды: сначала для одного массива, затем для другого.