Муниципальное автономное общеобразовательное учреждение

лицей №34 города Тюмени

ТЕМА УРОКА

«РЕКУРСИЯ. РЕКУРСИВНЫЕ АЛГОРИТМЫ»

10 класс

Вяткина Н.Л., учитель информатики

2021-2022 учебный год

**Тип:** урок открытия новых знаний, обретения новых умений и навыков.

**Цели:**

1. Деятельностная: научить новым способам нахождения знания, ввести новые понятия, термины.
2. Содержательная: сформировать систему новых понятий, расширить знания учеников за счет включения новых терминов, описаний, алгоритмов

**Задачи урока:**

**Предметные:**  формирование представления о рекурсивном объекте и рекурсивном алгоритме; освоение приемов реализации рекурсивных алгоритмов на языке Pascal.

**Метапредметные:**

**-** умение самостоятельно планировать пути достижения целей; умение соотносить свои действия с планируемыми результатами, осуществлять контроль своей деятельности, определять способы действий в рамках предложенных условий, корректировать свои действия в соответствии с изменяющейся ситуацией; умение оценивать правильность выполнения учебной задачи.

**Личностные:** - алгоритмическое мышление, необходимое для профессиональной деятельности в современном обществе; представление о программировании как сфере возможной профессиональной деятельности.

**Решаемые учебные задачи:**

1. напомнить сущность понятия вспомогательного алгоритма, подпрограммы;
2. напомнить способы реализации вспомогательных алгоритмов в Pascal
3. познакомить с примерами использования вспомогательных алгоротмов для реализации рекурсивных процедур.

**Педагогическая технология:** проблемного обучения.

**Метод:**словесный, наглядный, практическая работа.

**Формы работы:** фронтальная, выполнение практической работы, работа в группах.  
**Обеспечение урока:** доска, компьютерный класс, мультимедийный проектор, ОC Windows, среда программирования Free Pascal.

**Ожидаемые результаты:** Учащиеся должны

Знать:

* понятие «вспомогательный алгоритм», «рекурсивный алгоритм», «формальные и фактические параметры», «локальные и глобальные переменные»
* способы реализации вспомогательных алгоритмов, синтаксис и семантику их записи.

Уметь:

* осуществлять оптимальный выбор вспомогательного алгоритма;
* обосновать необходимость применения вспомогательного алгоритма;
* разрабатывать программы с применением рекурсивных алгоритмов.

**Структура урока:**

1. Организационный момент (1 мин.)

Приветствие, проверка присутствующих. Организация учебной работы.

1. Мотивационный этап. (5 мин.)

* Тестирование on-line <https://onlinetestpad.com/ru/test/13981-vspomogatelnye-algoritmy-podprogrammy>
* Анализ результатов

1. Открытие нового знания (10 мин.)
2. Первичное закрепление нового знания. Выполнение задания практической работы (12 мин)
3. Включение в систему знаний и умений. (10 мин.)
4. Рефлексия учебной деятельности, самоанализ (5 мин.)
5. Домашнее задание. Итог урока (2 мин.)

**Ход урока**

1. Организационный момент

Приветствие, проверка присутствующих. Организация учебной работы.

1. Мотивационный этап.
   1. Обучающиеся проходят индивидуальное on-line тестирование на сайте <https://onlinetestpad.com/ru/test/13981-vspomogatelnye-algoritmy-podprogrammy>
   2. Получение результатов тестирование, анализ ошибок.
2. Открытие нового знания

|  |  |
| --- | --- |
| **Учитель**: | **Учащиеся:** |
| Может ли процедура или функция содержать вызов других процедур или функций? | Выдвигают свои версии, приводят примеры. |
| А может ли процедура или функция вызывать сама себя? | Варианты ответов учеников |
| Сегодня мы и познакомимся с такими алгоритмами. | Записывают тему урока «Рекурсивные алгоритмы»  Совместно с учителем формулируют цели урока |
| Знакомит обучающихся с понятиями «рекурсия», «рекурсивный алгоритм» (Приложение 1) | Запись основных понятий в тетрадь |
| ***Решение задачи***  Используя рекурсивный алгоритм, вывести на печать цифры заданного натурального числа ***ch*** в обратном порядке.  Например, для входного числа 1598 на экран будет выведено 8951. (Приложение 2) | Учащимся предлагается распечатка программы решения задачи. По ходу объяснения учителя учащиеся заполняют таблицу исполнителя для рекурсивного алгоритма |

1. Первичное закрепление нового знания. Выполнение практической работы. Анализ результатов по эталону.
2. Включение в систему знаний и умений. Самостоятельная работа учащихся в группах. (Приложение 3)

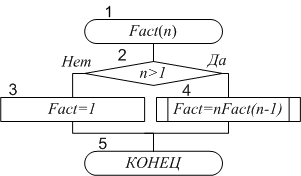
В качестве второго примера рассмотрим блок-схему вспомогательного алгоритма вычисления факториала числа

http://smiuk.sfu-kras.ru/kodnyanko/site/1_2_courses/AlgLabs/5/fact1.png

Нетрудно заметить, что http://smiuk.sfu-kras.ru/kodnyanko/site/1_2_courses/AlgLabs/5/fact2.png

Таким образом, значение факториала числа можно вычислить через факториал на единицу меньшего числа, то есть вычислить последующее через предыдущее и, следовательно, решить данную алгоритмическую задачу можно при помощи \_\_\_\_\_\_\_\_\_\_\_\_\_\_ алгоритма (рекурсивного).

Рассмотрим блок-схему вспомогательного алгоритма-функции **Fact**  для вычисления факториала числа **n**



Записать фрагмент вспомогательного алгоритма-функции.

1. Рефлексия учебной деятельности, самоанализ.

Ответы обсуждаются, при необходимости работа программы, вызвавшая затруднение, разбирается на доске с помощью учителя.

1. Домашнее задание. (Выучить конспект. Записать программу для вычисления факториала заданного числа.) Итог урока (2 мин.)

**Приложение 1**

Слово «рекурсия» происходит от латинского слова «recursio» - возвращение. Рекурсивные определения как мощный аналитический аппарат используется во многих областях науки, особенно в математике: арифметическая прогрессия (an+1=an+d); геометрическая прогрессия (bn+1=bnq, ); числа Фибоначчи, каждый элемент которого является суммой двух предыдущих; факториал (n!=1\*2\*3\*4\*5…).

***Рекурсия*** – это такой способ организации вычислительного процесса, при котором процедура или функция в ходе выполнения составляющих ее операторов обращается *сама к себе*. При использовании рекурсии необходимо обращать особое внимание *на выход из подпрограммы в нужный момент*, чтобы не вызвать зацикливания программы.

Рекурсивностью в Паскале могут обладать как функции, так и процедуры.

Таким образом, если в теле процедуры (функции) встречается вызов самой этой процедуры (функции), то это и есть рекурсия.

***Аналогия рекурсивного алгоритма:***

Матрешка, которая внутри содержит другую матрешку

Зеркальная комната с многократным отражением.

**Приложение 2**

**Идея решения задачи**

*Основная программа*: вводится натуральное число и передается во вспомогательный алгоритм.

*Вспомогательный алгоритм:* выделяется и печатается на экране младшая цифра числа. Если после усечения младшей цифры число не равно 0, то повторно вызывается вспомогательный алгоритм-процедура, но теперь без последнего (младшего) разряда. Таким образом, на каждом шаге число «теряет» младший разряд.

Рекурсивный алгоритм прекращает свою работу, когда число становится равным 0.

|  |  |
| --- | --- |
| *Program Z1;* | *Заголовок главной программы* |
| *var*  *ch: integer;* | *Раздел описания переменных* |
| *procedure rever(n:integer);*  *begin*  *write(n mod 10);*  *if (n div 10) <>0*  *then rever (n div 10);*  *end;* | *Вспомогательный алгоритм-процедура (ВАП)*  *N – формальный параметр*  *Выделение и печать младшей цифры числа*  *Если число не равно 0, то повторное обращение к ВАП* |
| *{основная программа}*  *begin*  *write(‘ введите натуральное число ‘);*  *readln ( ch);*  *rever(ch);*  *readln;*  *end.* | *Ввод числа*  *Обращение к ВАП. ch* – фактический параметр |

Заполним таблицу исполнителя для ВАП

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Шаг | *n* | *n mod 10* | *(n div 10) <>0 ?* | *rever ( )* |
| ОП |  |  |  | *Rever (1598)* |
| 1 | 1598 | 8 | 159 *<>0 Да* | *Rever (159)* |
| 2 | 159 | 9 | 15 *<>0 Да* | *Rever (15)* |
| 3 | 15 | 5 | 1 *<>0 Да* | *Rever (1)* |
| 4 | 1 | 1 | 0 *<>0 нет* |  |

Таким образом, из основной программы только одно обращение ко вспомогательному алгоритму, а во вспомогательном алгоритме процедура rever (для заданного числа) вызывает сама себя еще три раза.

Первый запуск процедуры выводит цифру 8, второе обращение с фактическим параметром 159, третье – с параметром 15, четвертое – с параметром 1. Завершаются процедуры в обратном порядке: четвертая передает управление третьей, третья – второй и т.д. , первая процедура завершается и передает управление в основную программу.

**Приложение 3**

{функция}

function fakt(n:integer):longint;

{описание функции, n –формальный параметр,}

{ результат выполнения функции имеет тип longint}

begin

if n=1

then fakt:=1 {проверка завершения рекурсии}

else fakt:=n\*fakt(n-1); {рекурсивное вычисление n!}

end;