Управление образования

Администрации Сергиево-Посадского района

Муниципальное бюджетное общеобразовательное учреждение

«Физико-математический лицей»

**«Как работает рекурсия. Понятие стека»**

Комбинированный урок, в 10 классе

Учитель: Перлова Н..В.

2018-2019 уч.г.

1. Общие сведения:

ФМЛ, 10 класс;

раздел «Алгоритмизация и программирование»

номер урока – 11.

учебные часы – 2.

1. Вводно-мотивационный этап:

**цели**:

приобретение учащимися навыков работы программирования рекурсивных подпрограмм;

развивающая: формирование приемов логического мышления, развитие интереса к предмету;

воспитательная: воспитание аккуратности, точности.

1. Ядро содержания обучения: изучение приема программирования - рекурсии.
2. Предварительная подготовка учащихся: изучение темы «Рекурсия», изученный материал на предыдущих уроках информатики.
3. Предварительная подготовка учителя: изучение материала урока, написание конспекта, разработка методических пособий.
4. Дидактические основания урока:

методы обучения: объяснительно-иллюстративный;

тип урока: комбинированный;

формы учебной работы учащихся: фронтальная работа, самостоятельная работа.

1. План урока:
2. Орг. Момент (1-2 мин)
3. Актуализация опорных знаний (10 мин)
4. Объяснение нового материала ( 15 мин)
5. Выполнение задания (15)
6. Итог урока (2 мин).

Ход урока: Рекурсия – обращение к самому себе в более упрощенном виде. Самый простой вариант увидеть рекурсию – это поставить зеркало напротив другого зеркала. Таким образом, что в зеркале отображается зеркало, и отражает другое зеркало, получится что-то вроде замкнутого цикла. В итоге мы будем наблюдать нечто похожее на тоннель. Рекурсия достаточно распространённое явление, которое встречается не только в областях науки, но и в повседневной жизни. Например, в стихотворении Самуила Маршака

Вот дом,

Который построил Джек.

А это пшеница,

Которая в темном чулане хранится

В доме,

Который построил Джек.

А это веселая птица-синица,

Которая часто ворует пшеницу,

Которая в темном чулане хранится

В доме,

Который построил Джек.

Вот кот,

Который пугает и ловит синицу,

Которая часто ворует пшеницу,

Которая в темном чулане хранится

В доме,

Который построил Джек.

Вот пес без хвоста,

Который за шиворот треплет кота,

Который пугает и ловит синицу,

Которая часто ворует пшеницу,

Которая в темном чулане хранится

В доме,

Который построил Джек.

А это корова безрогая,

Лягнувшая старого пса без хвоста,

Который за шиворот треплет кота,

Который пугает и ловит синицу,

Которая часто ворует пшеницу,

Которая в темном чулане хранится

В доме,

Который построил Джек.

В программировании рекурсия тесно связана с подпрограммамифункциями, точнее именно благодаря функциям в программировании существует такое понятие как рекурсия или рекурсивная функция. **Простыми словами, рекурсия – определение части функции (метода) через саму себя, то есть это функция, которая вызывает саму себя, непосредственно (в своём теле) или косвенно (через другую функцию).**  Типичными рекурсивными задачами являются задачи: нахождения n!, числа Фибоначчи. Такие задачи мы уже рассматривали, но решали их с использованием циклов. Вообще говоря, любой цикл можно решить рекурсивно, с использованием рекурсивной подпрограммы.

**Для описания рекурсивного решения необходимы две четко определенные вещи:**

1. **правило, по которому решение задачи в сложном случае сводится к решению такой же задачи, но в более простом случае;**
2. **условие, при котором дальнейшее упрощение нужно прекратить (терминальное условие).**

Разработаем программу, в которой объявлена рекурсивная функция, вычисляющая сумму цифр целого числа. Например, у нас есть число x=345329. Число может содержать любое количество разрядов. Выделить единицы мы можем всегда, не зависимо от количества разрядов. e=x%10/. Таким образом, можно сказать, что сумма цифр числа равна сумме последней цифры и сумме цифр оставшегося числа. Оставшееся число мы тоже можем найти x=x/10. Наша задача свелась к нахождению суммы цифр нового числа, с количеством разрядов на один меньше. Мы имеем:

1. *Найти цифру числа (9)*
2. *Сложить её с суммой цифр оставшегося числа(34532)*
	1. *Найти цифру числа (2)*
	2. *Сложить её с суммой цифр оставшегося числа (3453)*

Мыполучили такую же задачу, но в более простом виде. Мы вывели правило, по которому решение задачи в сложном случае сводится к решению такой же задачи, но в более простом случае.

Теперь нам надо определить терминальное условие, т. е. условие, когда дальнейшее упрощение нужно прекратить. В нашем случае, когда останется последняя цифра, т. е. если x<10. Получаем:

*Если число >10 то*

1. *Найти цифру числа*
2. *Сложить её с суммой цифр оставшегося числа.*

Мы написали тело рекурсивной подпрограммы. Теперь реализуем это на языке программирования:

*#include<iostream>*

*usingnamespacestd;*

*int sum(intn)*

*{*

 *int s = n % 10;*

 *if (n>= 10) s = s + sum(n / 10);*

 *return s;*

*}*

*intmain()*

*{*

 *setlocale(0, "Russian"); int a;*

 *cout<<"введитечисло"<<endl;*

 *cin>> a;*

 *cout<< sum(a);*

 *system("Pause");*

 *return 0;*

*}*

**Согласно правилам языка С++ — нельзя рекурсить функцию main.**

Теперь попробуем написать программу нахождения числа x в степени k, где k- целое положительное.

(Решение у доски учеником).

1. правило, по которому решение задачи в сложном случае сводится к решению такой же задачи, но в более простом случае: xk=x\*xk-1
2. терминальное условие: если k=0то xk=1

Реализация на языке:

*#include<iostream>*

*usingnamespacestd;*

*float step(floatx,intk)*

*{*

 *float s;*

 *if (k<1) s=1;*

 *else*

 *s=x\*step(x,k-1);*

 *return s;*

*}*

*intmain()*

*{*

 *setlocale(0, "Russian");*

 *cout<<"введитечислоистепень"<<endl;*

 *inta;float x;*

 *cin>> x>>a;*

 *cout<< step(x,a);*

 *system("Pause");*

 *return 0;*

*}*

Теперь разберемся, как работает рекурсия?

Вернемся к задаче нахождения суммы цифр числа. Для того чтобы лучше понять как работает программа вставим дополнительный вывод на экран:

int sum(intn)

{

 cout<<"n="<<n<<endl;

 int s;

 s = n % 10;

 if (n>= 10) s = s + sum(n / 10);

 cout<<"n="<<n<<endl;

 returns;

}

Выполним трассировку при n=123.

n =123

s=3

n(123)>10, условие выполняется, следовательно s=3+ далее идет вызов sum(12). Мы должны вернуться к началу выполнения подпрограммы. Встает вопрос, как хранить то что мы уже вычислили(s=3) и как узнать где мы прервались. Для решения этой задачи необходимо в памяти компьютера выделить место, где мы будем это хранить. Эту область имеет свое название – стек.

**Стек – область памяти, в которой хранятся локальные переменные и адреса возврата.**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |

Один из регистров процессора называется указателем стека – в нем записывается адрес последней занятой ячейки стека. При вызове подпрограммы в стек помещается значения всех её параметров, адрес возврата и выделяется место под локальные переменные.

Вызываем подпрограмму **sum(123)**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 123 | А | знач |  |  |  |  |  |  |  |

Где А – адрес возврата в программу после выполнения подпрограммы, знач – значение которое вернет подпрограмма. При вызове подпрограммы снова идет вызов подпрограммы **sum(12**). Имеем

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 123 | А123 | знач | 12 | 3 | А12 | знач |  |  |  |

Вызываем sum(1)

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 123 | А123 | знач | 12 | 3 | А12 | знач | 1 | 2 | A1 | знач |

**Самоанализ урока**

*Предмет*: информатика

*ФИО учителя:* Перлова Н.В.

*Тема урока*: **Как работает рекурсия. Понятие стека.**

*Дата*  03.10.18 *Класс 10*

*Тип урока*: комбинированный

**цели**:

образовательная:

* получение представления о реализации рекурсивного алгоритма в элктронно - вычислительной машине: понятие стека, хранение адресов данных в стеке,
* приобретение учащимися навыков работы программирования рекурсивных подпрограмм;

развивающая: формирование приемов логического мышления, развитие интереса к предмету;

воспитательная: воспитание аккуратности, точности.

**Методы и приемы***:*

Рассказ, использование мультимедийных средств, фронтальная работа учащихся, самостоятельная работа учащихся.

**Технологии**: ИКТ (*Создание печатных дидактических материалов.*

*Создание презентации*), энергосберегающие (*смена деятельности*), педагогика сотрудничества (*объяснение материала, самоконтроль учащихся*), проблемное обучение (*постановка преподавателем учебно-проблемной задачи, создание для учащихся проблемной ситуации; осознание, принятие и разрешение возникшей проблемы, в процессе которого учащиеся овладевают обобщенными способами приобретения новых знаний; применение данных способов для решения конкретных систем задач)*

**Краткое содержание урока***:* Актуализация опорных знаний (понятие рекурсии, правило описания рекурсивного решения), Объяснение нового материала, реализация рекурсивного алгоритма в элктронно - вычислительной машине: понятие стека, хранение адресов данных в стеке на примере решения задач нахождение суммы цифр числа. Самостоятельное выполнение трассировки рекурсивных алгоритмов.

**Выводы***:* Задачи урока успешно решаются благодаря тому, что его структура включает двухкратное рассмотрение основных вопросов учебного материала, при этом учитель варьирует методы изучения и формы работы учащихся. В начале урока учащиеся получают новые знания, затем закрепляют полученные знания (выходят к доске и записывают трассировку рекурсивного алгоритма ) и наконец, на уроке проводится самостоятельная работа учащихся с последующим взаимоконтролем, т.е. работа учащихся в группах . Данный приём развивает внимание, воспитывает ответственность учащихся за выполненную работу.