|  |
| --- |
|  |
| *«Алгоритми-процедури та алгоритми-функції»* |
| Конспект уроку. 11-А клас. 12.02.13р |
| **Студент групи Ф - 51****Бондарчук Костянтин** |
|  |
|  |

|  |
| --- |
|  |

**Вид уроку**: Комбінований; Тривалість: 45 хв.

**Матеріально-технічне забезпечення** : Персональні комп’ютери;

**Дидактичне забезпечення**: картки

**Програмне забезпечення**: Операційна система Windows ХРТема уроку: « Процедури і функції в мові програмування Формальні та фактичні параметри"

**Мета уроку**: Дати поняття про функції та процедури, оформлення підпрограм (процедур та функцій) мовою програмування Паскаль, поняття про формальні та фактичні параметри, області дії змінних.

Навчити використовувати на практиці здобуті знання при складанні програм.

Розвивати логічне мислення та творче уявлення.

Виховувати уважність та відповідальність.

**Обладнання**: Комп’ютери, картки

**Тип уроку**: урок засвоєння нового

І. Організаційна частина……….....................................................5 хв.

1.1. Перевірка наявності учнів

**ІІ. Основна частина……………………………………………….30 хв.**

**2.1. Повідомлення теми і навчальної мети уроку.**

**2.2. Актуалізація опорних знань учнів:**

Ми з вами складаючи програми уже частково знайомилися з основними стандартними функціями і процедурами. Давайте нагадаємо їх. Показую картки, а діти називають функцію і її призначення.

**Функції:** abs(x)-модуль x; cos(x)-косинус x, sin(x)-синус x, ln(x)-логарифм x, sqrt(x)-корінь квадратний x, sqr(x)-піднесення до квадрату x.

**Процедури:**inc(x,y) – збільшує х на y, inc(x)- збільшує х на 1.

dec(x,y) – зменшує х на y, dec(x)- зменшує х на 1.

Потім розглядаємо приклади на друкованих картках. Показую картку а діти називають результат дії функції чи процедури. Картки.



**2.3. Засвоєння нових знань та умінь**

**2.3.1. Пояснення нового матеріалу:**

Досить часто при розв’язанні певної задачі її розбивають на підзадачі, і кожна підзадача розв’язується окремо. Наприклад знаходження площі поверхні призми: знаходять площу основи, площу бічних граней, а потім сумують знайдені площі.

Щоб полегшити собі роботу під час складання програми ми усю програму можемо також розбити на підпрограми. Запишемо, що ми називаємо підпрограмою.

**Підпрограмою називається іменована, логічно закінчена група операторів мови, яку можна викликати для виконання будь-яку кількість разів із різних місць програми.**У мові Паскаль для організації підпрограм використовуються процедури і функції.

**1) Процедура - це незалежна поіменована частина програми, призначена для виконання визначених дій. Вона складається з тіла і заголовка. За структурою її можна розглядати як програму в мініатюрі. Після однократного опису процедуру дозволяється викликати за іменем з наступних частин програми. Використання імені процедури в програмі називається викликом процедури. Ім'я процедури не може знаходитися у виразі у якості операнду.**

Опис **процедури** включає заголовок (ім'я) і тіло процедури. Заголовок складається з зарезервованого слова **Procedure**, ідентифікатора (імені) процедури і необов'язкового списку формальних параметрів із вказівкою їх типу, який укладається в круглі дужки .

**Ім'я процедури** - ідентифікатор, унікальний у межах програми. Тіло процедури являє собою локальний блок, за структурою аналогічний програмі:

**Формат опису: (записую на дошці)**

*Procedure <ім'я> [(формальні параметри)];*

*[<розділи описів>;]*

*begin
<розділи операторів>*

*end;*

***Приклад:***
*Procedure Max (x,y:real);* **Пропоную учням навести приклади.**

У списку формальних параметрів перераховуються змінні разом із зазначенням їх типів.

Щоб звернутися до процедури, треба використати оператор виклику процедури. Він складається з ідентифікатора (імені) процедури і списку фактичних параметрів, що відділені один від одного комами і знаходяться у круглих дужках. Якщо процедурі не передається ніяких параметрів, то фактичні параметри не вказуються.

**Формат виклику процедури:**

*<ідентифікатор> [(фактичні параметри)];*

*Max(6,23);*

Параметри забезпечують механізм заміни, який дозволяє виконувати процедуру з різними початковими даними. Між фактичними параметрами в операторі виклику процедури і формальними параметрами у заголовку опису процедури встановлюється взаємо-однозначна відповідність у результаті їхнього перебору зліва направо. Фактичні параметри за кількістю і типами повинні дорівнювати кількості і типам формальних параметрів.

**2) Функція відрізняється від процедури тим, що, по-перше, передає в точку виклику скалярне значення (результат своєї роботи), а по-друге, ім'я функції може входити у вирази, як операнд. Функція, якщо вона зустрічається у виразі, називається покажчиком функції або звертанням до функції.**

**Функція**, визначена користувачем, складається з заголовка і тіла функції. Заголовок містить зарезервоване слово **Function**, ідентифікатор (ім'я) функції та, укладений у круглі дужки, необов'язковий список формальних параметрів і тип значення, що повертається функцією.

**Формат опису: ( записую на дошці)**

*Function <ім'я> [(формальні параметри)]:<тип результату>*

*[<розділи описів>;]*

*begin
<розділ операторів>*

*end;*

***Ім'я функції*** - унікальний у межах блока ідентифікатор.

Результат, що повертається, може мати будь-який простий тип і тип *string.*

Тіло функції являє собою локальний блок, за структурою аналогічний програмі.

У розділі операторів повинен перебувати хоча б один оператор, що присвоює ідентифікатору функції значення. Якщо таких операторів декілька, то результатом виконання функції буде значення останнього оператора присвоювання.

Звертання до функції здійснюється за іменем з необов'язковою вказівкою списку аргументів. Кожен аргумент повинен відповідати формальним параметрам, зазначеним у заголовку, і мати той же тип.

**Формат звертання:**

*Y:=<ідентифікатор функції > [(фактичні параметри)];*

*Y:=Max(a);*

Функції можуть повертати значення цілих, дійсних, булівских, літерних і рядкових типів.
Зверніть увагу, що як формальні параметри, так і розділ описів у процедурі може бути відсутній.

Усі процедури і функції мови Паскаль підрозділяються на дві групи:
 1)**вбудовані;**

**2)визначені користувачем.**

**Вбудовані (стандартні) процедури** і функції є частиною мови і можуть викликатися за іменем без попереднього опису в розділі описового блока. З багатьма з них ви вже знайомилися в попередніх розділах.

**Процедури і функції користувача** організовуються самим програмістом відповідно до синтаксису мови і являють собою локальні блоки. Попередній опис процедур і функцій користувача є обов'язковим.

**Змінні, описані в розділі описів основної програми, називаються *глобальними.***

Вони діють у всіх підпрограмах, з яких складається програма.

**Змінні, описані в розділі описів конкретної процедури, називаються *локальними.***

Вони діють тільки в межах даної проце­дури.

Процедури можуть отримувати і повертати значення не тільки через параметри-результати, але й через глобальні змінні. Тому списків параметрів у процедурі може і не бути.

**Параметри**, за допомогою яких здійснюється обмін значеннями змінних між підпрограмами та програмою, що їх викликає, можуть мати будь-який тип, в тому числі структурований.

 Існують два типи параметрів:

**параметр-значення;
параметр-змінна.**
 ***Група параметрів, перед якими відсутнє зарезервоване слово Var називається параметрами-значеннями.***
Наприклад, в описі *Procedure Max (x,y:real); x* і *y* - параметри-значення.

Формальний параметр-значення обробляється, як локальна стосовно процедури або функції, змінна. Зміни формальних параметрів-значень не впливають на відповідні значення фактичних параметрів.

***Група параметрів, перед якими знаходиться ключове слово Var називається параметрами-змінними.***

Наприклад, в описі *Procedure Obr(Var A,B:integer); А* та *В* - параметри-змінні. Параметр-змінна використовується в тому випадку, якщо значення повинно бути передане з процедури в блок, що її викликає. При активізації процедури або функції формальний параметр-змінна заміщується фактичною змінною, а тому будь-які зміни в значенні формального параметра-змінної відбиваються на фактичному параметрі.

І в тому, і в іншому випадку тип фактичного параметра повинен збігатися з типом формального. Якщо формальний параметр має рядковий тип, йому надається атрибут довжини, рівний 255, а тому і фактичний параметр в цьому випадку повинен також мати рядковий тип з атрибутом довжини, що дорівнює 255. У якості параметра-змінної може використовуватися будь-який тип, в тому числі файловий.
**Область дії ідентифікаторів.**

Для правильного визначення області дії ідентифікаторів при використанні в програмі процедур і функцій необхідно притримуватися наступних правил:

1. Кожний ідентифікатор повинен бути описаний перед тим, як він буде використаний.
2. Ідентифікатор діє у межах блоку, в якому він описаний.
3. Всі ідентифікатори в одному блоці повинні бути унікальними, тобто не повторюватися.
4. Однакові ідентифікатори можуть бути по-різному визначені у кожному окремому блоці, але це вважається поганим стилем програмування і тому не рекомендується в різних блоках програми використовувати змінні з однаковими іменами..
5. Якщо ідентифікатор підпрограми користувача збігається з ім'ям стандартної процедури або функції, то вони стають недоступними в межах області дії підпрограми, оголошеної користувачем, тобто стандартна функція ігнорується, а виконується програма користувача.

**3. Закріплення нового матеріалу…………………………10**

*Приклад.* Створимо функцію для обчислення *tg(x)* та обчислимо значення виразу tg(x)+ctg(x)+tg2(x).

 **program** Myfunc;

uses Crt;

**var** x,y:real;

 **function** tg(x:real):real;

 **begin**

tg:=sin(x)/cos(x);

**end;**

 **begin**

writeln(‘BBeдіть x');

readln(x);

y:=tg(x)+l/tg(x)+sqr(tg(x));

writeln('y=', y:5:2);

readln;

end.

Наприкінці уроку рекомендуємо перевірити засвоєння матеріалу на запитаннях для

самоконтролю. Наприклад:

1. Що таке підпрограма?
2. Чим відрізняються процедури від функцій?
3. Формати опису процедур і функцій.
4. Що таке локальні та глобальні параметри? Яка між ними відмінність?
5. Які параметри називаються фактичними, а які формальними?
6. Змінні яких типів можуть використовуватись в якості параметрів процедур та функцій?
7. Чи можуть існувати підпрограми без формальних параметрів?
8. Чим відрізняються параметри-значення від параметрів-змінних?
9. Що можна ставити у відповідність формальним параметрам-значенням, а що параметрам-змінним?

Крім того, можна запропонувати учням деякі тестові завдання, наприклад, такого типу.
З**адача 1.**
***Умова:*** Знайдіть і поясніть помилки в записі функцій:

**Function** max(n:integer):real;

**Var** a,max:real;

**Begin**

**Read**(max);

**For** i:=1 **to** n-1 **do**

**Begin**

**Read**(a);

**If** a>max **then** max:=a;

**End**;

**End**;

***Очікувана відповідь учнів:***

* У функції описана внутрішня локальна змінна max, ім'я якої співпадає з іменем функції, а це неприпустимо, тому що вони обидві являються локальними для даної функції і не можуть мати однакові імена.
* Якщо все ж таки ім'я функції max, то неможливо використання його у операторах read(max) та if a>max, тому що ми отримуємо самовиклик функції, а це може призвести до помилки.
* В операторах read(max) та if a>max помилка, якщо max - ім'я функції, тому що після імені функції в момент її виклику повинні знаходитись в дужках фактичні параметри, кількість та тип яких мають співпадати з кількістю та типом фактичних параметрів даної функції (в даному випадку фактичний параметр повинен бути один).
* Якщо в програмі не існує глобальна змінна і, то вона залишається неописаною в підпрограмі.

***Домашнє завдання:***

* + Вивчити конспект уроку
	+ Розв’язати задачі №1-№6
* Розв’язати задачі 1-6 використовуючи окремі процедури та функції
* 1. Радіус Місяця 1740 км. Обчислити площу поверхні S=4Пr2 та об'єм планети V=(4/3)Пr3.
* 2. Обчислити довжину гіпотенузи та площу прямокутного трикутника за заданими двома катетами.
* 3. Обчислити об'єм та площу бічної поверхні куба, якщо відоме ребро.
* 4. Яку площу і периметр матиме квадрат, описаний навколо круга заданої площі S*.*