**Складання програм з використанням модулів.**

Теоретичні відомості.

Прогресивним підходом у програмувонні вважається використання модулів або блоків. Набір процедур та функцій, об’єднаних в один блок (UNIT), може компілюватися незалежно від головної програми. Завдяки цьому час компіляції для великих програм може на багато зменшуватися, а це має велике значення при відладці програм, які приходиться досить часто компілювати.

МОДУЛЬ(UNIT) – це програмна одиниця, текст якої відкомпільовано незалежно. Вона включає в себе визначення констант, типів даних, змінних, процедур та функцій, доступних для використання в програмах, які викликають.

Внутрішня структура модуля прихована від користувача.

Структура модуля.

В модулі можна виділити декілька розділів: заголовок, інтерфейсна частина, реалізаційна частина і інілізаційна частина.

Заголовок модуля.

UNIT ім’я модуля;

Інтерфейсна частина.

INTERFACE початок розділу оголошень;

USES модулі, які використовуються;

LABEL оголошення доступних глобальних

міток;

CONST оголошення доступних глобальних

констант;

TYPE оголошення доступних глобальних

типів;

VAR оголошення доступних глобальних

змінних;

PROCEDURE(FUNCTION) заголовки доступних процедур (функцій);

Реалізаційна частина.

IMPLEMENTATION початок розділу реалізації;

USES модулі, які використовуються при реалізації;

LABEL оголошення скритих глобальних міток;

CONST оголошення глобальних констант;

TYPE оголошення скритих глобальних типів;

VAR оголошення скритих глобальних змінних;

PROCEDURE(FUNCTION) тіла доступних та скритих процедур (функцій);

Інілізаційна частина.

BEGIN основний блок модуля.

END.

Індивідуальне завдання.

Варіант №7.

Скласти програму на мові Pascal розв’язку даної задачі та виконання її на ЕОМ. Процедури та функції оформити у вигляді модуля.

Задача 1.

Модуль:

unit gregory;

{---------------------------------------------------------}

interface

uses crt;

var

 c,k:array [1..5,1..4] of real;

 i,j:integer;

 m:byte;

function facktorial(x:integer):real;

{---------------------------------------------------------}

implementation

function facktorial(x:integer):real;

var

 i:integer;

 facktorial1:real;

begin

 facktorial1:=1;

 if i<=1 then facktorial:=1 else begin

 for i:=1 to x do facktorial1:=facktorial1\*i;

 facktorial:=facktorial1;

 end;

end;

{---------------------------------------------------------}

begin

end.

Запускна програма:

uses gregory;

begin

clrscr;

 writeln('Введiть елементи масива:');

 writeln;

 for i:=1 to 5 do

 for j:=1 to 4 do begin

 write('Елемент [',i,',',j,']=');

 readln(c[i,j]);

 end;

 m:=0;

 clrscr;

 for i:=1 to 5 do begin

 for j:=1 to 4 do begin

 k[i,j]:=c[i,j]/facktorial(i+j);

 if k[i,j]<1 then m:=m+1;

 write(k[i,j]:7:5,' ');

 end;

 writeln;

 end;

end.

Робота з файлами.

Теоретичні відомості

Файл – це сукупність даних однакового типу. Таке визначення збігається з визначенням масиву, але, на відміну від останнього, файл зберігають на диску, кількість даних у файлі при його описуванні не вказується, елементи файла не мають індексів, що ускоаднює роботу з ними.

Опис файлового типу має вигляд:

 Type <ім'я типу> = File of <тип компонент>;

Процедури і функції для роботи з файлами:

Assign – зв'язує деякий зовнішній файл на диску з файловою змінною.

Reset – відкриває уже раніше створений файл данних на диску, ім'я якого було перед цим пов'язане процедурою Assign.

Rewrite – створює новий файл і відкриває його. Якщо файл вже існує вміст знищується, а сам файл відкривається знову.

Close – закриває файл, який зв'язаний з файловою змінною.

Eof – набуває значення true, якщо досягнуто кінець файла, інакше набуває значення false.

IOResult – використовується для пошуку помилок, які виникають при роботі з файлами.

Індивідуальне завдання

Задача 1.

Написати програму контролю знань студентів з літератури.

Оформити файл відповідей та файл питань.

Задача 2.

Написати програму що виводить на екран студентів, що набрали прохідний бал не нижче 8.

Задача 1:

program zavd1;

uses

 crt;

const

 qfile='quest.txt';

 afile='ansver.txt';

var

 f1,f2:text;

 i,k:integer;

 name,ansv:string;

begin

 clrscr;

 assign(f1,qfile);

 assign(f2,afile);

 rewrite(f2);

 reset(f1);

 write('Введіть своє ім`я, групу :');

 readln(name);

 writeln(f2,name);

 while not eof(f1) do begin

 readln(f1,name);

 writeln(name);

 write('Ваша відповідь :');

 readln(name);

 writeln(f2,name);

 readln(f1,ansv);

 if ansv=name then k:=k+1;

 i:=i+1;

 end;

 writeln(f2,'Всього питань :');

 writeln(f2,i);

 writeln(f2,'Правильних питань :');

 writeln(f2,k);

 close(f1);

 close(f2);

end.

РЕЗУЛЬТАТИ РОБОТИ ПРОГРАМИ:

Вміст файлу протоколу:

Бригіда e02

толстой

поет

да

Всього питань :

3

Правильних питань :

2

Задача 2:

program typefiles;

uses

 crt;

const

 kolocenok=2;

type

 abiturient=record

 fio:string;

 marks:array [1..kolocenok] of integer;

 end;

var

 ser,i,j,k:integer;

 abit: array [1..40] of abiturient;

 temp: abiturient;

begin

 clrscr;

 write('Введiть кiлькiсть абiтурiентiв: ');

 readln(k);

 for i:=1 to k do begin

 writeln('абiтурiент ',i,':');

 write('im`я: ');

 readln(temp.fio);

 writeln('Введiть оцiнки: ');

 for j:=1 to kolocenok do readln(temp.marks[j]);

 abit[i]:=temp;

 end;

 clrscr;

 for i:=1 to k do begin

 writeln('абiтурiент ',i,':');

 write('im`я: ',abit[i].fio);

 write(' мае оцiнки: ');

 ser:=0;

 for j:=1 to kolocenok do begin

 write(abit[i].marks[j],', ');

 ser:=ser+abit[i].marks[j];

 end;

 writeln;

 if ser>8 then writeln(' З А Ч И С Л Е Н И Й ! ');

 writeln;

 end;

end.

РЕЗУЛЬТАТИ РОБОТИ ПРОГРАМИ:

абiтурiент 1:

im`я: Саша мае оцiнки: 4, 5,

ЗАЧИСЛЕНИЙ!

абiтурiент 2:

im`я: Петро мае оцiнки: 3, 4,

абiтурiент 3:

im`я: Олег мае оцiнки: 4, 2,

абiтурiент 4:

im`я: Микола мае оцiнки: 2, 2,

Висновок:поглибив знання, вміння та навички розробки з використанням текстових та типизованих файлів, закріпив свої знання в області записів та типизованих файлів.