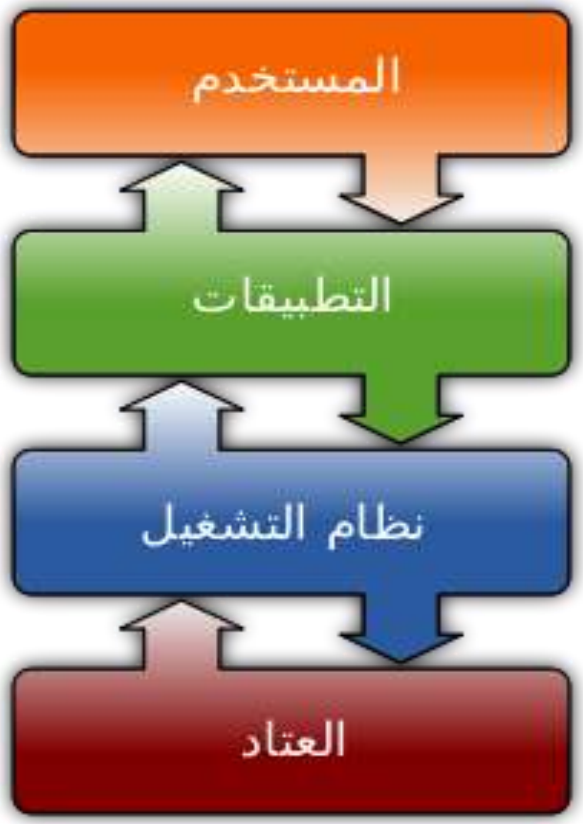
# جامعة حماة
# كلية العلوم في مصياف / السنة الأولى




لغات البرمجة

- **المادة:** [redacted] و [redacted]
- **المحاضرة** [redacted] السابعة

**الملفات في لغة باسكال**

File s Structure of Pascal

- **اسم المدّرس : د. م. هيثم وطفة**

# مثال ١ عن السجلات Example

- لدينا شركة تحتوي على مجموعة من العمال و الموظفين عددهم لا يتجاوز (١٠٠) مائة، و لنفرض لدينا سجل الموظف Employ يتألف من عدة حقول:
  - الاسم الثلاثي Name
  - تاريخ الولادة Birth Date
  - الجنس Sex
  - المهنة Profession
  - الراتب Salary
  - العنوان Address
- المطلوب كتابة برنامج يقوم بإدخال بيانات الموظفين المذكورة أعلاه تمهيداً لتنفيذ العمليات التالية عليها:

١- حساب عدد الموظفين الذكور الذين راتبهم الشهري (٥٠٠٠ خمسة الآف ليرة ) وما فوق.

٢- طباعة قائمة بأسماء الموظفات الإناث اللاتي مهنتهن " سكرتيرة " ؟

```
Program Employer ( input, output );
Type    n max = 1 .. 100;
        Employ = record
        Name : string[50];
        BirthDate : 1970 .. 2100;
        Sex: (Male, Female);
        Profession : String [ 20 ];
        Salary : Real;
        Address : String [ 50 ];
        End;
   Emp = array [ n max ] of Employ;
Var
    Em: Emp;
    n : n max ;
    i , SumEmp : Integer;
```

```
Begin
    Write ( ' Please Enter the Number of
    Employers n = ' );      Readln ( n );
    For i = 1 to n do
    Begin
    Write (' Enter Name := ' );
          readln (Em [ i ]. Name);
    Write (' Enter Year of BirthDate :=');
          readln (Em [ i ].BirthDate);
    Write (' Enter Sex of Employer :=');
          readln ( Em [ i ].Sex );
    Write (' Enter Profession :=');
          readln ( Em [ i ].Profession);
    Write (' Enter Salary :=');
          readln ( Em [ i ].Salary );
    Write (' Enter Address :=');
          readln ( Em [ i ].Address );
    End;
```

## حل الطلب الأول

```
SumEmp = 0 ;
    Writeln ('Number Of Male
    Employers & Salary more
    5000 SP');
    For i =1 to n do
    Begin
    If  (Em [ i ].Sex = ' Male ') and
    (Em [ i ].Salary ≥ 5000.0) then
    SumEmp = SumEmp + 1;
    End;
Writeln ( ‘SumEmp =‘, SumEmp );
```

## حل الطلب الثاني

```
Writeln ('Names Of Female
    Secretariat');
    For i =1 to n do
    Begin
    If  (Em [ i ].Sex = ' Female ') and
    (Em [ i ].Profession = ' سكرتيرة')
    then
    Writeln ( i ,' – ',Em [ i ].Name);
    End;
 End.
```

# مثال ٢ عن السجلات Example

- يبيع معمل الألبان في مدينة حمص منتجاته الغذائية إلى مجموعة من الزبائن بحيث لا يتجاوز عدد عمليات البيع (١٠٠٠) عملية، يود مدير المعمل بتخزين بياناته في قاعدة بيانات حاسوبية، لنفرض لدينا سجل مبيعات الزبائن Customer يتألف من عدة حقول:
  - **رقم الفاتورة Num**
  - **تاريخ العملية Date**
  - **اسم الزبون Name**
  - **عنوان الزبون Address**
  - **رقم هاتف الزبون TelNum**
  - **اسم المادة المباعة Product**
  - **الكمية Quant**
  - **سعر المادة Price**

- **المطلوب:** كتابة برنامج يقوم بإدخال بيانات الزبائن المذكورة أعلاه تمهيداً لتنفيذ العمليات التالية عليها:

١- حساب عدد عمليات البيع للزبون "X" حتى تاريخه.

٢- طباعة جدول بعمليات البيع للزبون "Y" يحتوي على اسم الزبون، رقم الفاتورة، تاريخ العملية، اسم المادة، الكمية، السعر، قيمة الفاتورة Total،القيمة الإجمالية للعمليات Sum.

٣-طباعة جدول بأسماء الزبائن يحتوي على أسمائهم و عناوينهم و أرقام هواتفهم ؟

```
Program Company ( input, output );
Type     n max = 1 .. 1000;
Customer = record
    Num, Quant, Telnum : Integer;
    D :Date;
    Name, Product : String [ 30 ];
    Price : Real;
    Address : String [ 50 ];
    End;
Cus = array [ n max ] of Customer;
Var
    Cu: Cus;
    n : n max ;
    i , SumX : Integer;
    x, y: String [25];
     Total: Real;
```

```
Begin
    Write ( ' Please Enter the Number of
    Operations n = ' );        Readln ( n );
    For i = 1 to n do
    With Cu [ i ] do
    Begin
Write ('Name := ' );      readln (Name);
Write ('Num :=');          readln (Num);
Write ('Date :=');          readln (Date);
Write ('Product :=');   readln (Product);
Write ('Quant :=');        readln (Quant);
Write ('Enter Price :=');  readln (Price );
Write ('Address :=');  readln (Address );
Write ('Telnum :=');      readln (Telnum);
    end;     {Begin}
    end;     {With}
```

## حل الطلب الأول

```
Sum = 0 ;
    Writeln ('Input X name');
          readln (X);
    For i =1 to n do
    With Cu [ i ] do
    If  (Name = X) then
    Sum := Sum + 1;
    End;
    Writeln ( 'Total of X Customer  =', Sum );
```

## حل الطلب الثاني

```
SumY=0;
    Writeln ('Input Y name'); readln (Y);
    For i =1 to n do
With Cu [ i ] do
    Begin
    If  (Name = Y) then
          Begin
          Write (Name);
          Write (Num);
          Write (Date);
          Write (Product);
          Write (Quant);
          Write (Price );
          Total : = Quant * Price ;
          Writeln ('Total =', Total );
          SumY= SumY+(Total );
          end;    {Begin}
    Writeln (' Total SumY = ', SumY);
    end;   {Begin}
end;       {With}
```

# حل الطلب الثالث

```
Writeln ('Table of Customer ');
    For i =1 to n do
With Cu [ i ] do
    Begin
    Write (Name);
    Write (Address );
    Writeln (Telnum );
    end;            {Begin}
end;                {With}
End.                {Begin}
```

# بنية الملفات File Structure

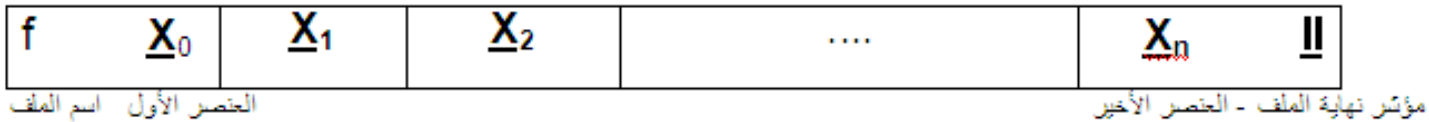
## ١- بنية الملف المتتالي Sequential File Structure:

**الملف**: هو متتالية عناصر من نوع قياسي واحد، أو بنيوي معرّف واحد، و عدد هذه العناصر غير محدد سلفاً. و لا يمكن الوصول إلى عنصر من عناصر الملف دون المرور بما قبله.

- كما يمكن تعريفه بأنه بنية معطيات متقدمة Advanced Structure تكون عناصره من نوع قاعدي واحد $T_0$ و يتكون الملف من عدد غير محدد من العناصر المختلفة و المرتبة من نوع واحد.
- يطلق على الملف اسم المتتالية، لأن التعامل معه يتم وفق القواعد المستخدمة عند التعامل مع المتتاليات الرياضية، و لبناء هذه المتتالية " الملف " يجب بناء أول عنصر فيه $X_0$ ومن خلال قيمته تحسب بقية القيم بالتتالي " مجموعة عناصر معرّفة بعلاقة تكرارية من الشكل $X_n = f ( X_{n-1} )$ ".
- يتم إنشاء الملف على أحد أقراص التخزين الحاسوبية كالقرص المغناطيسي الصلب HDD أو قرص الفلاش Flash Memory أو الشريط المغناطيسي Tape، باسم محدد يتم الإعلان عنه في بداية البرنامج و يكون هذا الملف خالياً بعد تعليمة الإنشاء rewrite ( x )، حيث يرمز للملف الخالي < > .

# بنية الملف المتتالي Sequential File Structure

-File الملف -




- يتم التعريف عنه بواسطة التعليمة Type على الشكل التالي في بداية البرنامج:

Type T = File of $T_0$
Var x : T ;

مثال:

```
Type     Text = file of char;
Type     Marks = file of integer;
Type     Names = file of String [45];
Var    t : Text ;
       M: Marks;
       N: Names;
```

- عند التعريف عن متحول من نوع ملفي قاعدي $T_0$ مثل ( t, M, N ) يتم تخصيص مكان في ذاكرة الحاسب يتسع لعنصر واحد فقط من الملف عند التصريح عن ملف في بداية البرنامج، و يتم الوصول إلى العنصر المراد معالجة محتوياته بعد أن يتم المرور على كافة العناصر التي تقع قبله " بالتتالي ".

# العمليات الأساسية المعرّفة على الملفات

- **يعرّف على الملفات " المتتاليات " مجموعة العمليات الأساسية التالية:**

١. **إنشاء ملف جديد اسمه x rewrite ( x ):** تتم عملية بناء متتالية خالية بالإجراء، فيقوم بعملية الإسناد التالية x =: < > ، كما تستخدم هذه التعليمة لمحو محتويات ملف ( x) تم إنشاؤه مسبقاً.

- تتوضع عبارة نهاية الملف eof في بداية الملف و يكون الملف جاهزاً للكتابة فيه.

٢. **دمج متتاليتين** من نوع قاعدي واحد $T_0$ في متتالية واحدة يتم على الشكل التالي:

X = < x1, x2, …,xn >

Y = < y1, y2, …, yn >

F = X & Y = < x1,x2, …, xn, y1, y2, …, yn >

٣. **First ( x )** :ترمز هذه الدالة إلى العنصر الأول في المتتالية ( x1 ).

X = < x1, x2, …,xn >

First ( x ) = ( x1 )

# العمليات الأساسية المعرّفة على الملفات

٤. **Rest ( x )** :ترمز هذه الدالة إلى متتالية تحوي جميع عناصر المتتالية x ماعدا العنصر الأول ( x1 ).

X = < x1, x2, …,xn >

Rest ( x ) = < x2, x3, …,xn >

X = < first (x) > & rest (x)

٥. **put ( x )**( تمديد متتالية): أي إضافة قيمة إلى نهاية الملف عندما يكون الملف مفتوح للكتابة، فيقوم بعملية الإسناد التالية x:= x & <x^>، حيث يضاف العنصر الجديد في نهاية الملف و تنتقل إشارة نهاية الملف مكاناً إلى الخلف و يبقى المؤشر دالاً عليها.

يستخدم نظام Turbo Pascal الإجراء write بدلاً من put ، حيث يتم التخلص من المتحول المؤقت x^ أثناء عملية الكتابة، طالما eof ( x ) = false و تستخدم على الشكل التالي:

Write ( x , e ); ↔ x^ := e; put ( x );

٦. **reset ( x )**( تنظيم عرض عناصر متتالية ): عند فتح ملف للقراءة يجب أن يتوقف مؤشر الملف عند أول عنصر في المتتالية " بداية الملف " تجهيزاً لقراءته، و يتم ذلك بواسطة تعليمة reset ( x )، فيقوم هذا الإجراء بالعملية التالية: x^ := first ( x ) .

# العمليات الأساسية المعرّفة على الملفات

٧. **get ( x )** : الانتقال من عنصر إلى العنصر التالي في ملف مفتوح للقراءة.

يستخدم نظام Turbo Pascal الإجراء Read بدلاً من get ، حيث يتم التخلص من المتحول المؤقت x^ أثناء عملية القراءة، طالما eof ( x ) = false و تستخدم على الشكل التالي:

```
Read ( x , v );    ↔        v := x^;  get ( x );
```

٨. **assign ( x , ' filename ' )** : ربط المتحول الملفي باسم الملف الخارجي المخزّن على القرص.

٩. **Close ( x )** : يجب إغلاق الملف عند انتهاء التعامل معه في حالتي القراءة و الكتابة.

١٠. **eof** (دالة الوصول إلى نهاية الملف ): عند قراءة ملف مفتوح للقراءة، يجب أن يكون لدينا إمكانية الكشف عن إشارة نهاية الملف، حيث تكون قيمة eof ( x ) = true عند الوصول إلى نهاية الملف، و طالما قيمة eof ( x ) = false يمكننا متابعة القراءة. تستخدم عادة في قراءة الملفات على الشكل التالي:

```
Reset ( x );
While not eof (x) do
        Begin
                ….
        end;
```

- **مثال ١**: اكتب برنامجاً يقوم بإنشاء ملف يحتوي على مربعات الأعداد الصحيحة من ١ إلى n ؟

```
Program Numbers ( input,x );
Type    Number = file of integer;
Var     x : Number;          {متغير الملف المؤقت}
        i , j , n : integer;
Begin                        {بداية البرنامج}
    Assign ( x , ' c: num.dat ' ); {ربط الملف}
    Rewrite (x);   { إنشاء ملف}
    Write (' Please Input n= ');
    Read ( n );
    while i < = n Do
          Begin
          j := sqr(i);
          write(x,j);
          i := i + 1;
          end;
    close (x); { إغلاق الملف}
End.      {نهاية البرنامج}
```

- **مثال ٢**: اكتب برنامجاً يقوم بطباعة الأعداد الزوجية فقط من الملف السابق الذي يحتوي على مربعات الأعداد الصحيحة من ١ إلى n ؟

```
Program Num2 (x,output);
Type    Number = file of integer;
Var     x : Number;
        i  : integer;
Begin   {بداية البرنامج}
    Assign ( x , ' c: num.dat ' ); { ربط الملف}
    Reset (x);      {الانتقال إلى بداية الملف}
    while not eof (x) Do      {التحقق من الوصول إلى نهاية الملف}
            Begin
            Read ( x,i );
            if i mod 2 = 0 then
            Writeln (i);
            end;
    close (x);      {إغلاق الملف}
End.    {نهاية البرنامج}
```

# وظيفة Homework

- ✓ اكتب برنامجاً لتحضير ملف School سجلات طلاب مدرسة بحيث يتألف السجل الواحد Student من عدة حقول:
  - ١- الاسم الأول First Name
  - ٢- أسم الأب Father Name
  - ٣- الكنية Family Name
  - ٤- تاريخ الولادة Birth Date
  - ٥- الصف و الشعبة Course

و يحوي السجل الأخير اسم الطالب الوهمي 'zzz'؟

- ✓ اكتب برنامج لطباعة أسماء صف معيّن مدخل من الملف المحضّر في المثال السابق ؟

# بنية الملفات File Structure

## ٢- الملفات النصية Text Files:

الملف النصي تكون محتوياته عبارة عن نصوص وغالبا ما يكون امتداده .txt

في الملفات النصية يمكن الكتابة والقراءة أيضا بعد إنشائه في مجلد البرنامج ، أي موجود في المجلد TPW داخل القرص C.

- **يجب ترتيب خطوات التعامل مع الملفات النصية:**

١- **تعريف متغير** من النوع النصي text (وليكن F).

```
Var
F: text;   {التعريف باسم متغير الملف}
```

٢- **تعيين assign** للمتغير F بملف نصي (موجود مسبقا، أو ملف نصي جديد) : أي ربط المتغير بإسم الملف ويكون بعد بداية البرنامج Begin. باعتبار الملف الموجود باسم File1.txt .

```
Begin
Assign (F,' File1.txt');      {ربط اسم الملف بالمتغير المعرّف}
```

# Text Files الملفات النصية

- عندما نريد أن ننشئ ملف نصي جديد (أي مستخدم البرنامج هو الذي يكتب الاسم ) يجب أن نعرّف متغير ما لكي يحفظ فيه اسم الملف مؤقتاً ريثما يتم ربطه بالأسم الذي يريده المستخدم وليكن S:

```
Var
  F: text;
  S: string;
    Begin
        Read(s);          {إدخال اسم الملف إلى المتغير س}
        Assign (f, s);    {ربط اسم الملف المخزّن في المتغير س بالمتغيّر f}
```

- ملاحظة: المتغير سيكون عبارة عن مجموعه من الحروف والأرقام، أي سلسله نصيه (لأن اسم الملف s من النوع string).

٣- **فتح الملف** النصي الموجود مسبقا:

```
Reset (f); {فتح الملف}
```

- أما في حاله إنشاء ملف جديد

```
Rewrite (f);  { نشاء الملف}
```

# Text Files الملفات النصية

٤- **قراءه البيانات** (في الملف القديم ) ويتم ذلك

**Readln (f, x);** {قراءة البيانات من الملف المخزّن ف و تخزينها في المتغيرx}

Writeln(x); {الطباعة على الشاشة لمحتوى المتغيرx}

- المتغير X يجب أن يكون من النوع string ومعنى هذه الجملة أن يقوم المترجم بقراءة السطر في الملف وتخزينه في المتغير X و من ثم طباعته المتغير X على الشاشة .
- أما في حالة إدخال البيانات لملف جديد

Readln(x);

Writeln (f, x);

أي قم بقراءة المتغير x (من النوع string )ومن ثم قم بطباعته في الملف F

٥- **التحقق من الوصول إلى نهاية الملف النصي** ، أي هل انتهى المترجم من قراءه الملف النصي ووصل إلى نهاية الملف

**Eof (f);** **{التحقق من الوصول إلى نهاية الملف}**

If eof (f) then writeln (‘The End’);

end of file هي اختصار لـ

٦- **إغلاق الملف النصي**

**Close (f);** {إغلاق الملف}

## مثال على قراءة ملف مخزّن مسبقاً في مجلد البرنامج باسم romansy.txt

```
Program ex1 (input, output);
Var
    f: text;
    s: string;
    Begin
    Assign (f,'romansy.txt');
    Reset (f);
    Repeat
          Readln (f, s);
          Writeln(s);
    Until eof (f);
    Close (f);
    End.
```

**شرح البرنامج :**

- في البداية نقوم بتعريف متغيرين، الأول f من النوع النصي text والثاني المتغير s من النوع string .
- نبدأ البرنامج بتعليمة begin ثم نقوم بربط assign الملف romansy.txt بالمتغير f .
- ثم نقوم بفتح الملف بالعبارة reset(f) وتبدأ بعدها حلقة repeat، أي نقوم بقراءة السطر الأول في الملف النصي ونقوم بتخزينه في المتغير s ثم بطباعة المتغير s ثم نصل إلى الجملة Until eof(f); فيقوم بالتحقق من الشرط (أي هل وصل الملف إلى نهايته) أم لا .
- إذا كان نعم (أي وصل) يقوم بالخروج من الحلقة.
- إذا كان لا فيقوم بتكرار هذه الحلقة حتى تتحقق الحلقة (أي وصل إلى نهاية الملف) .

## مثال على قراءة ملف مخزّن مسبقاً في مجلد البرنامج باسم romansy.txt

- الفائدة من الحلقة repeat قراءة الملف النصي كاملا من أوله إلى آخره وطبعاً في حالة الملفات النصية ، لا يمكن استخدام الحلقة for لأن عدد السطور غير معلوم ، لذا يجب أن نستخدم جملة تكرار تستخدم الشرط وهما while أو repeat.
- في المثال السابق ، لو أردنا أن يكون اسم الملف النصي هو الذي يدخله المستخدم ، يكون شكله كاللاتي:
- المتغير n هو متغير من النوع string

```
Begin
    Read (n);
    Assign (f, n);
    :
    :
End.
```

## مثال على إدخال بيانات إلى ملف جديد (أي يتم إنشاء ملف بأي اسم ومن ثم تتم كتابة بعض البيانات فيه ).

```
Program ex2 (input, output);
Var
    F: text;
    S, n: string;
    Ch: char;
Begin
    Read (n);
    Assign (f, n);
    Rewrite (f);
    Repeat
        Readln(s);
        Writeln (f, s);
        Ch: = readkey;
    Until ch =#27
Close (f);
End.
```

- **شرح البرنامج :**
- بعد تعريف المتغيرات F للملف النصي، n لاسم الملف المدخل من قبل المستخدم، s لطباعة البيانات المدخلة من المستخدم إلى الملف النصي، المتغير ch هو عبارة عن متغير حرفي char
- بدأ البرنامج ، ثم قمنا بقراءة المتغير n وعيناه assign إلى الملف النصي، و قمنا بإنشائه rewrite .
- بدأت حلقه التكرار، اقرأ المتغير s بواسطة تعليمة Readln(s) ومن ثم أطبعه على الملف النصي Writeln(f,s)
- ماذا يقصد بـ (Ch:= readkey;): هي عبارة عن داله تستخدم لقراءة (الحرف أو الرقم أو الرمز) المدخل من المستخدم وقمنا بتعينه (إسناده) إلى المتغير ch .
- سوف تتكرر هذه الحلقة (أي قراءه المتغير s و طباعته في الملف النصي) حتى تتحقق الحلقة Ch =# 27 حتى تساوي قيمة 27# ومعناه المفتاح Esc الموجود في أعلى لوحة المفاتيح من الجهة اليسرى . معنى ذلك سوف تكرر الحلقة حتى يقوم المستخدم بالضغط على مفتاح( Esc الهروب).

# الملفات النصية الثنائية Binary File

- **يوجد نوعين من الملفات الثنائية هما:**

١. **الملفات المطبوعة Typed File**: تعني أن الملف يحتوي صيغة بيانات واحدة بكافه محتوياتها، هذا النوع يتضمن قواعد البيانات لان كل محتوياتها سجلات بيانات (سجل ملفات).

٢. **الملفات غير المطبوعة Untyped File**: تعني أن الملف لا يحتوي على صيغة واحدة، هذا النوع يحتوي على معلومات اضافيه قد تكون تركيبا مختلفا للسجلات.

# ١- الملفات المطبوعة Typed File

- الخطوات التي نتبعها في الملفات المطبوعة Typed File تشبه الخطوات في الملفات النصية Text File:
- إسناد المتغير F باسم الملف وذلك بالعبارة assign
- فتح الملف reset
- إنشاء ملف جديد rewrite
- يجب تغير readln و writeln إلى read و write.
- إغلاق الملف close
- الفرق بين الملفات النصية و الملفات المطبوعة إذا قمت بفتح ملف باستخدام reset هذا لا يعني انك تستطيع القراءة منه فقط (كما في الملفات النصية)، بل يمكن اضافه المزيد وأيضا امكانيه تعديله .

- لنفترض بأنك عرفت سجل بهذا الشكل

```
Type
    Data = record
            Name: string [10];
            Age: integer;
            Tel: integer;
    End;
```

هذا السجل سيكون بالشكل Typed file

```
Var
    F: file of data;
```

# مثال على الملفات المطبوعة Typed File

```
Program example (input, output);
Type
    Data=record
    Name: string;
    Age: integer;
    End;
Var
    f: file of data;
    d: data;
    c: char;
    s: string;
Begin
    Write (‘Input File name: ‘);
    Readln(s);
    Assign (f, s);
    Rewrite (f);
```

```
Repeat
    Clrscr;
    Write (‘Name= ‘); readln (d.name);
    Write (‘Age = ‘); readln (d.age);
    Write (f, d);
    Write (‘Input New Data Y/N ‘);
    Repeat
    c: = upcase (readkey);
    Until c in [‘Y’,’N’];
    Write(c);
Until c=’N’;
Close (f);
End.
```