

**Кафедра Кріогенної та Мікроелектроніки**

**Обробка експериментальних даних за допомогою  
Турбо Паскалю**

## Вступ.

Експериментальні дані, що їх вимірює експериментатор, більшою частиною потребують попередньої обробки перед їхньою інтерпретацією. Шляхи обробки експериментальних даних залежать від їх подання. Часто обробку даних можна провести за допомогою вже існуючих пакетів програм, або загального призначення (таких, як табличний процесор *Microsoft Excel*, що має більш-менш розвинутий набір функцій математичної обробки даних) або спеціалізованих програм, призначених для обробки даних в різних галузях фізичного експерименту. Однак трапляються випадки, коли певна функція обробки відсутня в доступних користувачу пакетах, або її реалізація не задовольняє користувача. Тоді користувач, що має певні навички програмування, може реалізувати ці функції самостійно.

В лабораторній роботі ставиться **мета** реалізувати за допомогою мови програмування високого рівня деякі прості функції обробки експериментальних даних електронної спектроскопії. В цій роботі для програмування вибрана мова *паскаль*, але допускається користування іншою мовою високого рівня (в лабораторії доступні компілятори з *паскалю*, *сі* та *фортрану*).

Для обробки в цій роботі використовується реальний *електронний спектр*, знятий за допомогою однієї з експериментальних установок в Лабораторії Електронної Спектроскопії. *Електронний спектр* являє собою набір точок (відліків) деякої величини (в нашому випадку це похідна  $dN/dE$  від розподілу вторинних електронів  $N(E)$  за енергією цих електронів  $E$ ). Цей спектр поданий в вигляді текстового файлу типу:

```
3.00000E+0001 -6.95000E-0002
3.05000E+0001 -7.05000E-0002
3.10000E+0001 -7.60000E-0002
3.15000E+0001 -8.05000E-0002
3.20000E+0001 -8.40000E-0002
.....
```

тобто являє собою дві колонки чисел. Перша колонка — це енергії вторинних електронів (в електронвольтах), друга — значення похідної від розподілу по енергії  $dN/dE$ . Ми вважаємо, що дані зняти з постійним шагом по енергії (в нашому випадку шаг дорівнює 0,5). Оскільки шаг за енергією постійний, то повністю першу колонку з файлу зчитувати необов'язково, достатньо лише прочитати перше та останнє значення енергії в цій колонці та визначити число відліків в файлі).

**Завдання**, що ставиться в роботі — написати закінчену програму, що зчитує файл спектральних даних з диску, робить над ним певну операцію (див. варіанти завдань), після чого записує оброблені дані в інший файл. Після цього зобразити ці дані в графічному вигляді за допомогою програми, що дозволяє малювати

двовимірні графіки (можна користуватися програмами *Microsoft Excel*, *Grapher for Windows/DOS* та ін.) та роздрукувати їх.

У випадку програмування з використанням *Turbo Pascal* для вводу-виводу текстових файлів даних можна користуватися найпростішою бібліотекою вводу-виводу. Ця бібліотека знаходиться в файлі TXTDATIO.TPU (лістинг файлу TXTDATIO.PAS цієї бібліотеки поданий в додатку). Бібліотека включає в себе дві процедури — ReadDAT та WriteDAT, приклад використання яких поданий нижче.

### Приклад використання бібліотеки TXTDATIO.

```
program TestTXTIO;
{Програма тестування бібліотеки TXTDATIO}

uses TXTDATIO; {цей оператор вказує турбо-паскалю на необхідність}
               {використання бібліотеки TXTDATIO.TPU}

var {визначення змінних, що використовуються програмою}
    Data1, Data2: array[1..1000] of Double;
    {В масив Data1 буде поміщено спектральні відліки (тобто другу колонку
    з файлу даних. Double - це тип даних, подібний типу Real, але більш
    зручний для використання у розрахунках при наявності сопроцесору
    80x87. В масив Data2 будуть поміщені дані після обробки}
    X1, X2: Double; {це - значення початку та кінця шкали енергій,
    тобто перше та останнє значення першої колонки
    в файлі даних}
    N: Integer;    {кількість відліків в файлі даних}
    I: Integer;    {допоміжна змінна, що буде використована в програмі}

begin {початок програми TestTXTIO}
    ReadDAT('TEST.DAT', X1, X2, N, Data1); {читаємо дані з файлу 'TEST.DAT'}

    {як приклад, ми лише множимо вхідні дані на деяку константу (1.23)}
    for I := 1 to N do
        Data2[I] := Data1[I] * 1.23;

    WriteDAT('TEST1.DAT', X1, X2, N, Data2); {записуємо оброблені дані}
end. {кінець програми TestTXTIO}
```

Цю програму можна використати як шаблон для виконання завдання лабораторної роботи.

### Варіанти завдань.

1. Написати програму, що читає з диску файл TEST.DAT, згладжує дані по 3-м точкам (лінійне згладжування) та записує вихідний файл TEST1.DAT. Зробити те ж саме, але повторити процедуру згладжування кілька разів та порівняти результати.

**Лінійне згладжування** — це процедура, яка дозволяє зменшити рівень шумів в спектрі. Згладжений спектр

$N^{32l}(E)$  обраховується за вхідним спектром  $N(E)$  таким чином:

$$N_i^{32l} = \frac{\sum_{k=i-n}^{i+n} N_k}{2n+1}$$

для всіх точок спектру  $i$ . Кількість точок згладжування дорівнює  $2n+1$ , тобто для згладжування по трьом точкам треба взяти  $n=1$ , для 5-ти точок  $n=2$  ... Як видно, лінійне згладжування — це просто усереднення по найближчим точкам.

*Додаткове завдання.* Оскільки ми поліпшуємо один з параметрів нашого спектру, а саме зменшуємо рівень шумів, деякі інші параметри спектру повинні погіршуватися, бо кількість інформації залишилася тою ж самою. Які саме параметри погіршуються?

2. Написати програму, що читає з диску файл TEST.DAT, та знаходить інтеграл від спектру (зауважимо, що вхідні дані - це електронний спектр у вигляді  $dN/dE$ , тому на виході ми повинні одержати розподіл  $N(E)$ ). Записати дані в вихідний файл TEST1.DAT. Інтегрування можна провадити за методом трапецій. Наприклад, якщо треба обрахувати інтеграл від функції  $f(E)$   $\left(F(E) = \int f(E)dE\right)$  за методом трапецій, ми робимо так:

$$F_i = h \sum_{k=2}^i \frac{f_k + f_{k-1}}{2} \equiv F_{i-1} + h \frac{f_i + f_{i-1}}{2}, i \geq 2$$
$$F_1 = 0.$$

для всіх точок спектру  $i$ , причому вважається, що нумерація точок починається з одиниці. Тут  $h$  — шаг в спектрі.

### Література

1. Ю.С. Бородич, А.Н. Вальвачев, А.И. Кузьмич. Паскаль для персональных компьютеров. Справочное пособие. Минск, изд-во “Вышейшая Школа”, 1991.
2. Т. Рютген, Г. Франкен. Турбо Паскаль 6.0. Киев: “BNV”, 1992.
3. Описание лабораторной работы “Турбо паскаль и компьютерная графика”. Часть 2 и список литературы в нем.

## Додаток. Лістинг найпростішої бібліотеки вводу-виводу файлів даних.

```
{=====}
{= Найпростіша бібліотека вводу-виводу файлів даних.          =}
{= Ніякого контролю помилок вводу-виводу не провадиться.      =}
{= Вважається, що дані мають постійний шаг та енергія збільшується =}
{= Автор: П'ятницький М.Ю.                                     =}
{=====}

{$I+}
unit TXTDATIO;

interface

procedure ReadDAT(FileName: String; {ім'я файлу даних}
  var X1, X2: Double; {границі за енергією}
  var N: Integer;    {кількість відліків}
  var D);            {дані}
{-читає дані з файлу з ім'ям FileName. Використовуємо безтипову
змінну D, щоб забезпечити незалежність від розмірності масиву}

procedure WriteDAT(FileName: String; {ім'я файлу даних}
  X1, X2: Double; {границі за енергією}
  N: Integer;    {кількість відліків}
  var D);        {дані}
{-записує дані в файл з ім'ям FileName. Використовуємо безтипову
змінну D, щоб забезпечити незалежність від розмірності масиву}

implementation

type
  DummyType = array[1..$F000 div SizeOf(Double)] of Double;

procedure ReadDAT(FileName: String; {ім'я файлу даних}
  var X1, X2: Double; {границі за енергією}
  var N: Integer;    {кількість відліків}
  var D);            {дані}
{-читає дані з файлу з ім'ям FileName. Використовуємо безтипову
змінну D, щоб забезпечити незалежність від розмірності масиву}
var
  F: Text;
  Dymmy: DummyType absolute D;
  X: Double;
begin
  Assign(F, FileName);
  Reset(F);
  N := 0;
  while not SeekEOF(F) do begin {читаємо до кінця файлу}
    Inc(N);
    ReadLn(F, X, Dymmy[N]);
    if N = 1 then {Якщо перше значення - беремо початкове значення енергії}
      X1 := X;
  end;
  X2 := X;      {Останнє значення - кінцеве значення енергії}
  Close(F);
end; {ReadDAT}

{-----}

procedure WriteDAT(FileName: String; {ім'я файлу даних}
  X1, X2: Double; {границі за енергією}
  N: Integer;    {кількість відліків}
  var D);        {дані}
{-записує дані в файл з ім'ям FileName. Використовуємо безтипову
змінну D, щоб забезпечити незалежність від розмірності масиву}
var
  F: Text;
  Dymmy: DummyType absolute D;
  X: Double;
  I: Integer;
  Step: Double;
```

```
begin
  Assign(F, FileName);
  Rewrite(F);
  Step := (X2 - X1) / (N - 1);
  for I := 1 to N do begin
    X := X1 + Step * (I - 1);
    WriteLn(F, X:12, ' ', Dummy[I]:12);
  end;
  Close(F);
end; {WriteDAT}

end.
```