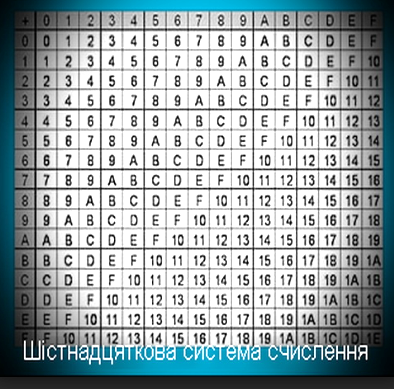
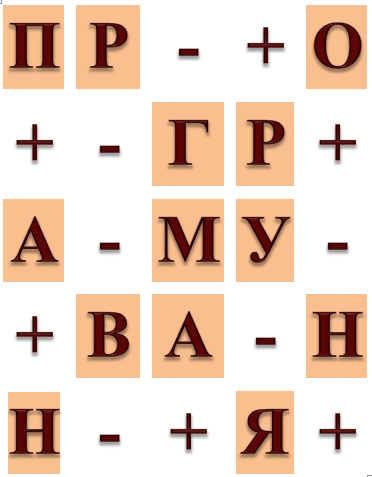
**Щасливський навчально виховний комплекс**

**Остапець Володимир Степанович**



**З чого почати**



**(*навчально-методичний посібник*)**

**2017 рік**

**ЗМІСТ**

|  |  |  |
| --- | --- | --- |
| 1 | Вступ | 3 |
|  |  |  |
| 2 | Відомості про системи числення та їх застосування | 5 |
|  |  |  |
| 3 | Перетворення десяткового числа в шістнадцяткове і навпаки | 9 |
|  |  |  |
| 3.1 | Перетворення десяткового числа в шістнадцяткове | 9 |
|  |  |  |
| 3.2 | Перетворення шістнадцяткового числа в десяткове | 10 |
|  |  |  |
| 4 | Перетворення двійкового числа в шістнадцядкову систему числення і шістнадцядкового числа у двійкову систему числення | 15 |
|  |  |  |
| 5 | Способи вводу/виводу параметрів у паскаль-програмах | 21 |
|  |  |  |
| 6 | Переведення чисел між довільними системами числення | 29 |
|  |  |  |
| 7 | Застосування методів багаторозрядної арифметики в програмах перетворення чисел між позиційними системами числення | 35 |
|  |  |  |
| 7.1 | Стандартний спосіб | 35 |
|  |  |  |
| 7.2 | Економний спосіб | 38 |
|  |  |  |
| 8 | Деякі “секрети” для юних програмістів | 39 |
|  |  |  |
| 9 | Висновки | 43 |
|  |  |  |
|  | Література | 45 |
| 10 | Додатки (програми мовою pascal для переведення чисел з однієї системи числення в іншу) | 46 |

**ВСТУП**

**1**

Людство завжди шукало шляхи вирішення проблем, тобто розв’язки задач. При цьому однією із таких проблем завжди була, є і в майбутньому буде проблема автоматизації знаходження розв’язків задач, тобто програмувати розв’яків. Адже мати програму завжди привабливіше, ніж шукати “вручну”. Та все тече і все змінюється, як сказав античний мудрець. Галузь програмування за півстоліття дуже змінилась. Колись програмувати було дуже важко, довго і нудно. Сьогодні ж воно, так би мовити, візуалізувалось, менш обтяжене трудоємкими діями, частково автоматизоване. Поряд з цим з’явилась дуже велика кількість вузьких спеціалізацій програмування, які важко відслідковувати. В той же час зросла спокуса обминути труднощі.

На протязі чотирьох десятиліть педагогічної практики спостерігались хвилі прояву збільшення та послаблення інтересу в учнів до програмування. Сьогодні цей інтерес близький до критичного нуля. Ради справедливості варто зауважити, що й концепція і стандарти сучасної освіти сьогодні тільки допомагають зниженню актуалізації програмування, як важливого елемента інтелектуального розвитку школярів. У шкільній програмі з інформатики домінує інформаційно-технологічна складова, на рівні стандарту ознайомленню з моделюванням та формуванню уявлень про алгоритми відводиться лише 5 уроків. Це означає, що програмування в школі зовсім не розглядається. Правда, в нових програмах з інформатики в 5-8 класах заплановано вивчення алгоритмів, але перевага надається середовищам зразка Scratch, користь вивчення яких досить сумнівна, якщо не шкідлива. Не тільки програмування, навіть інформатики немає ні в ЗНО, ні вступних іспитах чи співбесідах. На перших курсах у вищих навчальних закладах програмування вивчається фактично з нуля. Дивним дисонансом цьому виглядять олімпіади з програмування, предмета, що не вивчається в загальноосвітніх щколах.

Але алгоритмічна складова при вивченні основ наук не менш важлива, ніж логічна. Без них навчання може бути чисто дилетантським. Виходячи з того, що систематичне програмування не передбачено загальноосвітніми програмами з інформатики, тут буде розглянуто підходи до формування культури програмування тільки у якості спеціальної, факультативної дисципліни. Виділимо кілька рівнів:

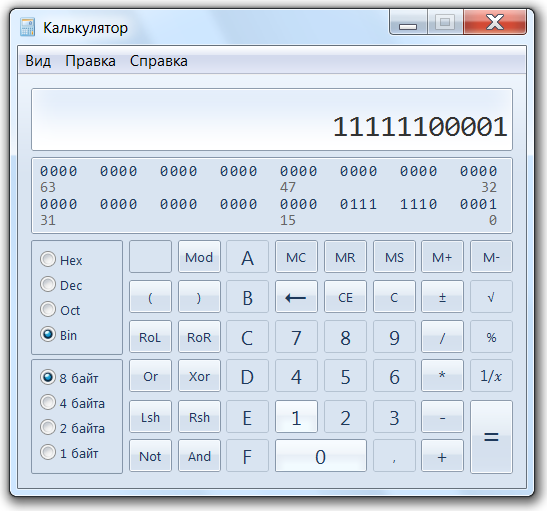
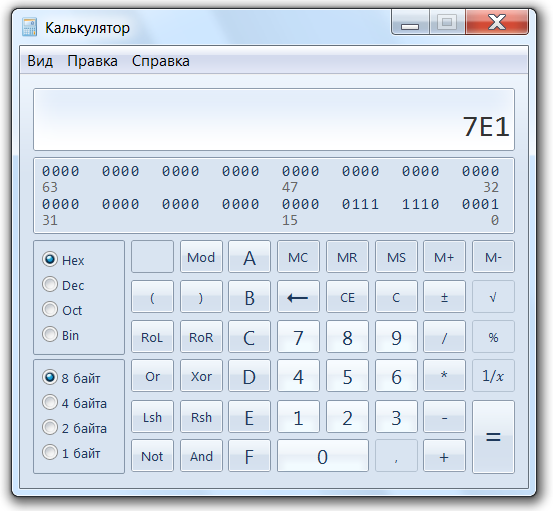
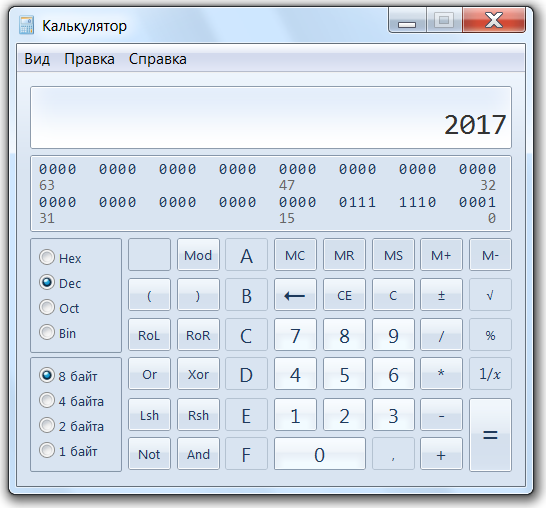
1. Ознайомлювальний рівень (формування уявлення про моделювання, алгоритмізацію та структурне програмування);
2. Рівень систематизації та формування техніки програмування;
3. Рівень проектування застосування спеціальних прийомів програмування.

Свого часу було опубліковано навчально-методичний посібник “Експрес-курс алгоритмізації та програмування” (ЕКАП), ряд методичних статей, зокрема, ˮАлгоритмічний підхід до розв’язування задач багаторозрядної арифметикиˮ, ˮМетод динамічного програмування в школіˮ та інші, тут основну увагу приділимо пункту 2), тобто виробленню техніки програмування.

Уроки (в наших умовах, про що було вище ‒ це завжди заняття, адже одне заняття може містити кілька споріднених уроків, бути розбите на кілька уроків або бути комбінацією класного заняття і самостійної домашньої роботи) з програмування завжди передбачають міжпредметні зв’язки інформатики та математики.

Будемо застосовувати метод поступового та поетапного ˮзануренняˮ в тему, суть якого полягає у прив’язці всього дослідження до однієї задачі, яку можна буде розвивати ˮв ширинуˮ та ˮв глибинуˮ, поступо розширюючи діапазон аналогічних задач та ускладнюючи кожну до категорії нетривіальних. Таким чином з’являється можливість всебічно дослідити конкретну вибрану тему, одночасно маючи велику кількість матеріалу для закріплення, а саме головне ‒ нагода ˮвідчутиˮ всі етапи програмування.

Як приклад, для розгляду візьмемо задачу переведення чисел з однієї системи в іншу. Тема широко розроблена, сьогодні можна знайти дуже багато матеріалів по ній в тому числі у інтернеті. У сучасних версіях Windows є навіть розширення ПРОГРАМІСТ стандартного додатку КАЛЬКУЛЯТОР (малюнок 1):



*мал. 1*

Скориставшись ним, миттєво можемо перевести натуральне число між десятковою, двійковою, восьмірковою та шістнадцятковою системами числення.

Отже, наведемо необхідні тут деякі відомості про системи числення і зокрема про позиційні системи числення.

**ВІДОМОСТІ ПРО СИСТЕМИ ЧИСЛЕННЯ**

**2**

**ТА ЇХ ЗАСТОСУВАННЯ**

Як відомо, існує багато систем числення, в тому числі **непозиційні**, **позиційні** та **змішані**.

У **непозиційних** системах числення величина, яку позначає цифра, не залежить від позиції її у числі. При цьому система може накладати обмеження на позиції цифр, наприклад, щоб вони були розташовані по спаданню, чи згруповані за значенням. Проте це не є принциповою умовою для розуміння записаних такими системами чисел.

Типовим прикладом непозиційної системи числення є [римська система числення](https://uk.wikipedia.org/wiki/%D0%A0%D0%B8%D0%BC%D1%81%D1%8C%D0%BA%D0%B0_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%86%D0%B8%D1%84%D1%80), в якій як цифри використовуються латинські букви:

|  |  |
| --- | --- |
| **Римська цифра** | **Десяткове значення** |
| I | 1 |
| V | 5 |
| X | 10 |
| L | 50 |
| C | 100 |
| D | 500 |
| M | 1000 |

Наприклад, VII = 5 + 1 + 1 = 7. Тут символи V і I означають 5 і 1, відповідно, незалежно від місця їх у числі.

У **позиційних** системах числення одна і та ж цифра (числовий знак) у записі числа набуває різних значень залежно від своєї позиції (звідси назва). Таким чином, позиція цифри має вагу в числі. Здебільшого вага кожної позиції кратна деякому натуральному числу яке називається основою системи числення.

Винахід позиційної системи числення приписують шумерам і вавілонцям. Її було розвинуто індусами і вона отримала неоціненні наслідки для історії людської цивилізації.

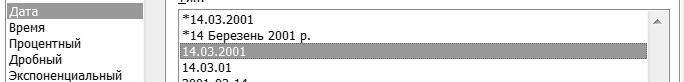
До числа таких систем належить сучасна десяткова система числення (з основою), виникнення якої пов'язують із лічбою на пальцях. У середньовічній Європі вона з'явилася через італійських купців, які у свою чергу запозичили її у мусульман.

**Змішана система числення** є узагальненням системи числення з основою і її часто відносять до позиційних систем числення. Основою змішаної системи є послідовність чисел, що зростає, кожне число представляється як [лінійна комбінація](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%BA%D1%82%D0%BE%D1%80%D0%BD%D0%B8%D0%B9_%D0%BF%D1%80%D0%BE%D1%81%D1%82%D1%96%D1%80).

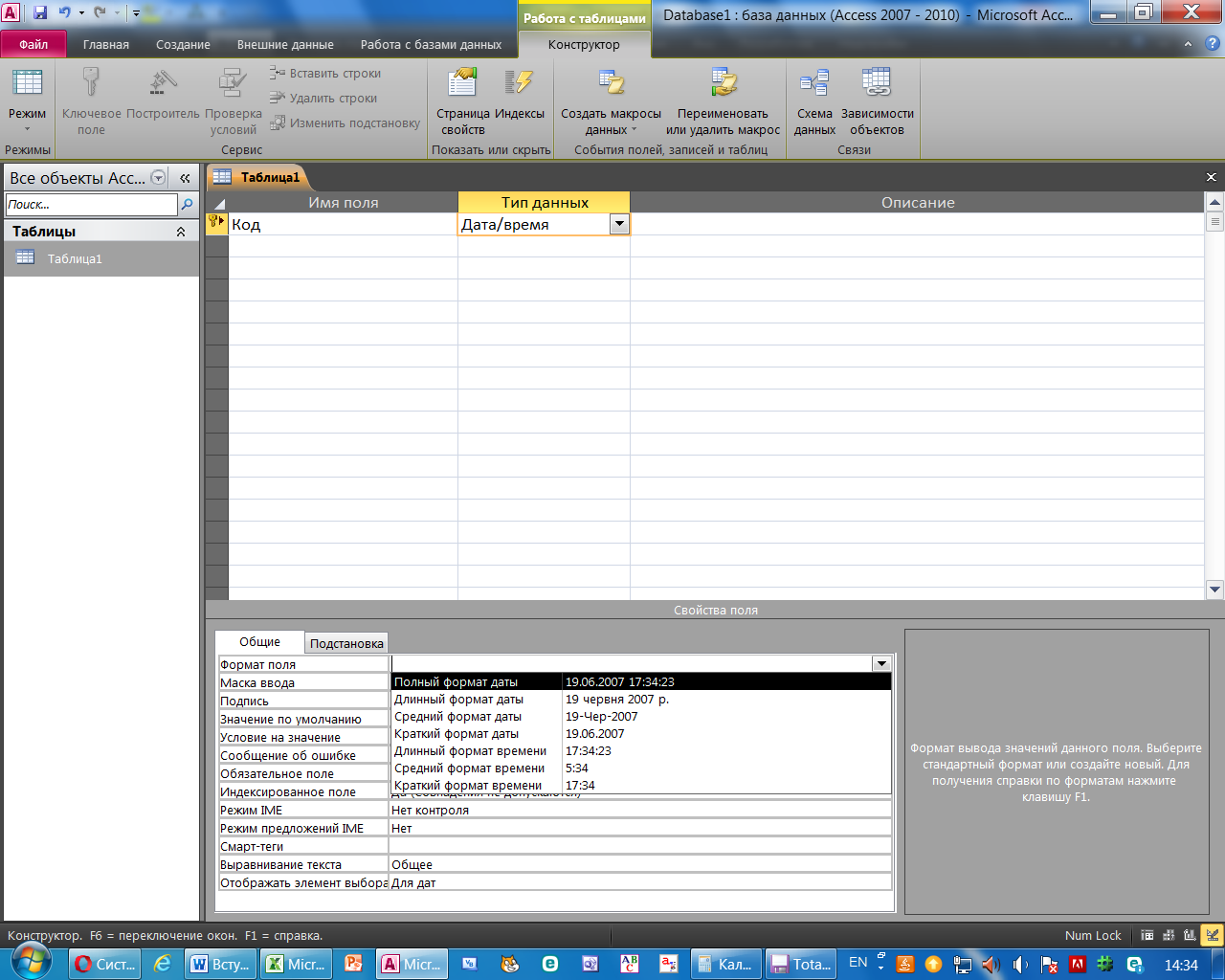
Найвідомішим прикладом змішаної системи числення є представлення часу у вигляді кількості діб, годин, хвилин і секунд. При цьому величини *d* (днів), *h (*годин), *m* (хвилин), *s* (секунд) відповідають значенням d⋅24⋅602 + h⋅602 +*m*⋅60 + s і розділяються між собою крапками, двокрапками, знаком “/” чи інакше, наприклад: 1⋅24⋅602 + h⋅602 +*m*⋅60 + s = 86400/50400/1500/12, що означає перше січня, 14 годин, 25 хвилин і 12 секунд.

Розвиток змішаної системи числення застосовується в Excel або Access при форматуванні чисел у вигляді дати/часу (див. малюнок 2).

У даній роботі не ставилась мета детального розгляду різних систем числення, тут ідеться про так звану комп’ютерну арифметику, тобто застосування систем числення в комп’ютерній техніці, в першу чергу **позиційних систем числення.**



Excel



Access

*мал. 2*

Як відомо в принципи реалізації комп’ютерної арифметики покладено позиційні системи числення: десяткова, двійкова, вісімкова та шістнадцяткова. Тим, хто вивчає комп’ютерні науки важливо розуміти правила та алгоритми переведення чисел між названими системами числення. Але узагальнення цих правил і алгоритмів може становити інтерес для програмістів-початківців, тобто бути окремою темою при вивченні програмування у школі.

У зв’язку з тим, що найчастіше здійснюється переведення десяткових чисел у систему з основою *k* (обмежимось *k* ≤ 16), переведення будемо проводити за схемою *k ↔* 10 ↔ *m*, де *k* та *m* ‒ основи систем числення (*k* ≤ 16, *m* ≤ 16). Тому спочатку зосередимось над складанням програм переведення чисел з десяткової системи у систему із основою *k* та навпаки, із системи числення з основою *k* в десяткову систему. Відповідні програми назвемо ***Num\_D\_k*** та ***Num\_k\_D***, де *k* ‒ основа вибраної системи числення, а *D* ‒ основа десяткової системи числення.

Така стратегія бачиться оптимальною, адже, як буде видно далі, програми ***Num\_D\_k*** та ***Num\_k\_D*** аналогічні, якщо змінювати *k*.

Зауважимо, що програми при *k <* 10 (першого виду) легко звести лише до опрацювання числових даних, а програми з основою *k =* *A*; *B*; *C*; *D*; *E* та *F* (другого виду) ‒ до одночасного використання як числових, так і текстових даних. Але для повної аналогії в програмних кодах будемо використовувати опрацювання поряд з числовими даними і текстових даних у всіх програмах, тобто другого виду. Почнемо із програм ***Num\_D\_16*** та ***Num\_16\_D***.

***Цікаві факти:*** Однією з відомих і поширених у старі часи систем числення була шестидесяткова. Вона виникла у шумерів в 3 тисячолітті до н. е., використовулась у стародавній Вавилонії. Зараз використовується в модифікованій формі для вимірювання часу, кутів, і географічних координат.

Нуль використовується лише всередині числа й ніколи не пишеться, коли в числі немає одиниць першого або першого й другого розрядів. Роль нуля відіграю пропуск.



*мал. 3*

Перший шістдесятковий знак після коми називається мінута (назва для часу: *хвилина*) ('), другий - секунда ("). Раніше використовувалися назви терція ( ‴ ) для третього знака, кварта (IV) для четвертого знака, квінта (V) для п'ятого знаку і т. д. Квадратний корінь з 2, ‒ довжина діагоналі одиничного квадрату, апроксимували вавилоняни старого вавилонського періоду (1900 до н.е.‒1650 до н.е.), як шістдесяткове число. {\displaystyle 1;24,51,10=1+{\frac {24}{60}}+{\frac {51}{60^{2}}}+{\frac {10}{60^{3}}}={\frac {30547}{21600}}=1.41421{\overline {296}}\approx 1.414212\ldots }Оскільки{\displaystyle {\sqrt {2}}} є ірраціональне число, то воно не може бути точно виражене в шістдесятковій системі числення, але його шістдесятковій запис починається так: 1;24,51,10,7,46,6,4,44…=

1 радіан ≈ 57°17′45″ = {\displaystyle 57+{\frac {17}{60}}+{\frac {45}{60^{2}}}} градусів.

Приблизне значення періоду повного обертання Землі навколо Сонця дорівнює 365; 15'24 "10 ‴ днів, приблизно 365,25671 днів.

***Запитання:***

1. Що називається системою числення?
2. Що таке позицій на система числення?
3. Що таке “вага” розряду в позиційних системах числення?
4. Що таке алфавіт системи числення?
5. Що таке двійкова система числення?
6. Які є правила переводу чисел з однієї системи чисел в іншу?
7. Що таке вісімкова система числення?
8. Що називається тріадою?
9. Що таке шістнадцяткова система числення?
10. Що називається тетрадою?
11. Що таке двійково-десяткова система числення?
12. Які до цього часу використовуються відомі системи числення?
13. До яких систем числення належать іонічна та кирилична системи числення?
14. Який вклад у математику зробив Кирик Новгородський?
15. В якій позиційній системі числення відсутня цифра “0”?
16. Що означає поняття “непослідовна позиційна система числення”?
17. Звідки походить термін “дюжина”?

***Завдання:***

1. У шестидесятковій системі числення для позначення використовувались цифри 1 ‒ (стоячий клин) ; 10 ‒ (лежачий клин). Відповідно числа: 3 ‒ **▼▼▼**; 32 ‒ **◄◄◄▼▼**. Часто позначки цифр об'єднувались, наприклад, число 32 записувалось так:

**Записати в шестидесятковій системі числення число 201710.**

1. У сучасній науковій літературі для зручності використовується компактний запис вавилонського числа, наприклад: 4,2,10; 46,52. Розшифровується цей запис наступним чином: 4 × 3600 + 2 × 60 + 10 + 46/60 + 52/3600. **Записати це число традиційним для вавілонської (шестидесяткової) системи числення**.

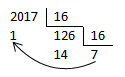
**П Е Р Е Т В О Р Е Н Н Я**

**3**

**ДЕСЯТКОВОГО ЧИСЛА В ШІСТНАДЦЯТКОВЕ І НАВПАКИ**

**3.1 ПЕРЕТВОРЕННЯ ДЕСЯТКОВОГО ЧИСЛА В ШІСТНАДЦЯТКОВЕ**

**АЛГОРИТМ Num\_D\_16** (переведення десяткового числа у шістнадцяткове**)**

1. перший етап:
2. другий етап:

201710 = 7·162 + 14·161 + 1·160 = 7·162 + E·161 + 1·160 = 7E116

**ПРОГРАМА**: *program Num\_D\_16;uses Crt;*

*var a16:string;aM,aD:integer;*

*begin*

*Write('десяткове число?');Read(aD);a16:='';*

*while aD>0 do begin {1}*

*aM:=aD mod 16;aD:=aD div 16; {2}*

*if aM=0 then a16:='0'+a16 else {2}*

*if aM=1 then a16:='1'+a16 else {2}*

*if aM=2 then a16:='2'+a16 else {2}*

*if aM=3 then a16:='3'+a16 else {2}*

*if aM=4 then a16:='4'+a16 else {2}*

*if aM=5 then a16:='5'+a16 else {2}*

*if aM=6 then a16:='6'+a16 else {2}*

*if aM=7 then a16:='7'+a16 else {2}*

*if aM=8 then a16:='8'+a16 else {2}*

*if aM=9 then a16:='9'+a16 else {2}*

*if aM=10 then a16:='A'+a16 else {2}*

*if aM=11 then a16:='B'+a16 else {2}*

*if aM=12 then a16:='C'+a16 else {2}*

*if aM=13 then a16:='D'+a16 else {2}*

*if aM=14 then a16:='E'+a16 else {2}*

*if aM=15 then a16:='F'+a16 end; {2}*

*WriteLn(a16)*

*end.*

|  |  |
| --- | --- |
| *N10* | *N16* |
| 2017 | 7E1 |
| 563 | 233 |
| 9278 | 243E |
| 16777164 | FFFFCC |
| 13954855 | D4EF27 |
| 2147483647 | 7FFFFFFF |

**ПРИКЛАДИ**:

**3.2 ПЕРЕТВОРЕННЯ ШІСТНАДЦЯТКОВОГО ЧИСЛА В ДЕСЯТКОВЕ**

**АЛГОРИТМ Num\_16\_D** (переведення шістнадцяткового числа у десяткове)

7E116 = 7·162 + E·161 + 1·160 = 7·256 + 14·16 + 1 = 201710

**ПРОГРАМА**: *program Num\_16\_D;uses Crt;*

*var a16,a:string;i,n:integer;aD,j,j16:real;*

*begin*

*Write('a16?');Read(a16);*

*n:=Length(a16);aD:=0;j:=0;*

*for i:=1 to n do begin {1}*

*a:=Copy(a16,n-i+1,1);jH:=Power(16,j);*

*if a='0' then aD:= aD+ 0\*j16 else {2}*

*if a='1' then aD:= aD+ 1\*j16 else {2}*

*if a='2' then aD:= aD+ 2\*j16 else {2}*

*if a='3' then aD:= aD+ 3\*j16 else {2}*

*if a='4' then aD:= aD+ 4\*j16 else {2}*

*if a='5' then aD:= aD+ 5\*j16 else {2}*

*if a='6' then aD:= aD+ 6\*j16 else {2}*

*if a='7' then aD:= aD+ 7\*j16 else {2}*

*if a='8' then aD:= aD+ 8\*j16 else {2}*

*if a='9' then aD:= aD+ 9\*j16 else {2}*

*if a='A' then aD:=aD+10\*j16 else {2}*

*if a='B' then aD:=aD+11\*j16 else {2}*

*if a='C' then aD:=aD+12\*j16 else {2}*

*if a='D' then aD:=aD+13\*j16 else {2}*

*if a='E' then aD:=aD+14\*j16 else {2}*

*if a='F' then aD:=aD+15\*j16 else {2}*

*j:=j+1 end;*

*WriteLn(aD);*

*end.*

|  |  |
| --- | --- |
| *N16* | *N10* |
| 7E1 | 2017 |
| 233 | 563 |
| 243E | 9278 |
| FFFFCC | 16777164 |
| D4EE27 | 13954855 |
| 7FFFFFFF | 2147483647 |

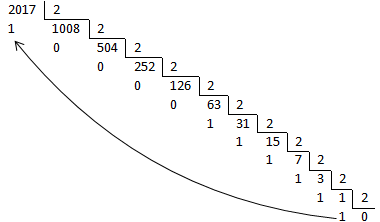
**ПРИКЛАДИ**:

**АНАЛІЗ**:

Проаналізуємо наведені програми. Для зручності у них використовуються однакові наіменування величин: aD ‒ значення в десятковій системі числення, a16 ‒ значення в шістнадцятковій системі числення. У звʹязку з тим, що у якості цифр шістнадцяткових чисел використовуються великі літери латиниці A ‒ 10; B ‒ 11; C ‒ 12; D ‒ 13; E ‒ 14; F ‒ 15, у розглядуваних програмах необхідний блок команд, що перекодовують текстові символи в десяткові числа і навпаки (позначено коментарем {2}). Цикл {1} забезпечує сам процес перекодування.

Покажемо, як трансформуються програми ***Num\_D\_16*** та ***Num\_16\_D*** у програми ***Num\_D\_2*** та ***Num\_2\_D***. Програми працюють в числовому діапазоні типу integer (до 214748364710 або 11111111111111111111111111111112).

**АЛГОРИТМ Num\_D\_2** (переведення десяткового числа у двійкове**)**

1. перший етап:
2. другий етап:

201710 = 1·210 + 1·29 + 1·28 + 1·27 + 1·26+ 1·25 + 0·24+ 0·23 + 0·22 + 0·21 + 1·20 =

= 1·1024 + 1·512 +1·256 + 1·128 + 1·64 + 1·32 + 0·16 + 0·8 + 0·4 + 0·2 + 1 =

= 111111000012.

**ПРОГРАМА***: program Num\_D\_2;uses Crt;*

*var a2:string;aM,aD:integer;*

*begin*

*Write('десяткове число?');Read(aD);a2:='';*

*while aD>0 do begin {1}*

*aM:=aD mod 2;aD:=aD div 2;*

*if aM=0 then a2:='0'+a2 else*

*if aM=1 then a2:='1'+a2 else*

*if aM=2 then a2:='2'+a2 else*

*if aM=3 then a2:='3'+a2 else*

*if aM=4 then a2:='4'+a2 else*

*if aM=5 then a2:='5'+a2 else*

*if aM=6 then a2:='6'+a2 else*

*if aM=7 then a2:='7'+a2 else*

*if aM=8 then a2:='8'+a2 else*

*if aM=9 then a2:='9'+a2 end;*

*writeLn(a2);*

*end.*

Корисно програму ***Num\_2\_D*** корисно співставити з наведеною вище програмою ***Num\_16\_D***.

**АЛГОРИТМ Num\_2\_D** (переведення двійкового числа у десяткове)

111111000012  = 1·210 + 1·29 + 1·28 + 1·27 + 1·26 + 1·25 + 0·24 + 0·23 + 0·22 + 0·21 +

+ 1·20 = 201710

**ПРОГРАМА***: program Num\_2\_D;uses Crt;*

*var a2,a:string;i,n:integer;aD,j,j2:real;*

*begin*

*Write('a2?');Read(a2);*

*n:=Length(a2);aD:=0;j:=0;*

*for i:=1 to n do begin*

*a:=Copy(a2,n-i+1,1);j2:=Power(2,j);*

*if a='0' then aD:= aD+0\*j2 else*

*if a='1' then aD:= aD+1\*j2 else*

*if a='2' then aD:= aD+2\*j2 else*

*if a='3' then aD:= aD+3\*j2 else*

*if a='4' then aD:= aD+4\*j2 else*

*if a='5' then aD:= aD+5\*j2 else*

*if a='6' then aD:= aD+6\*j2 else*

*if a='7' then aD:= aD+7\*j2 else*

*if a='8' then aD:= aD+8\*j2 else*

*if a='9' then aD:= aD+9\*j2;*

*j:=j+1 end;*

*WriteLn(aD) end.*

Програма ***Num\_D\_2*** може виглядіти значно коротше і привабливіше (назвемо ***Num\_D\_2\_Dubl***):

*program Num\_D\_2\_Dubl;uses Crt;*

*var a2:string;aD,i,aM:integer;*

*begin*

*Write('десяткове число?');Read(aD);*

*while aD>0 do begin {1}*

*aM:=aD mod 2;aD:=aD div 2;*

*for i:=0 to 9 do {2}*

*if c=i then a2:=IntToStr(c)+a2 end;*

*WriteLn(a2);*

*end.*

Проте, слід наголошувати учням на те, що коротша за виглядом програма не завжди більш ефективна, адже у програмі важливо намагатись дотримуватись мінімізації кількості величин, що викристовуються та підвищення швидкодії. Інколи виграючи в одному доводиться програвати в іншому. Звернемо увагу, що програма ***Num\_D\_2*** містить тільки один цикл {1}, а програма ***Num\_D\_2\_Dubl*** натомість два ({1} та {2}), причому, цикл {2} вкладений у цикл {1}. У достатньо громіздких програмах це може привести до виходу за часові рамки допустимого виконання програми. Крім того, програма ***Num\_D\_2\_Dubl*** має додаткову величину i:integer, що може в окремих випадках збільшити обсяг використаної пам’яті. Для наших прикладів це несуттєві зауваження, адже програми прості і не вимагають великої кількості арифметичних та логічних операцій. Та саме на простих і наочних програмах (інакше кажучи, ˮпрозорихˮ) саме й час та нагода формувати дуже потрібні юному програмісту так звані ˮправила хорошого тону в програмуванняˮ. Нарешті, у програмі ***Num\_D\_2\_Dubl*** використовується стандартна функція ***IntToStr***, яка може бути відсутня в інших версіях мови програмування ***Pascal***.

У програмі ***Num\_D\_2*** є ще дві, на перший погляд непомітні й дріб’язкові переваги. Якщо в ній замінити команди розгалуження скороченою формою (без *else*), то за рахунок повного перебору всіх команд розгалуження (тоді вони перестають бути вкладеними одна в одну) залежно від значення величини *aM* може бути суттєве збільшення часу виконання програми за рахунок збільшення довжини ітерацій.

Легко помітити, що програму ***Num\_2\_D*** теж можна скоротити з точки зору довжини програмного коду (пропонується виконати це самостійно).

Підводячи проміжний підсумок, зауважимо, що з огляду на наведені вище міркування тут свідомо вибрано варіант програми ***Num\_D\_2***.

В аналізі до програм ***Num\_D\_16*** та ***Num\_16\_D*** вже наголошувалося, що у всіх програмах зразка ***Num\_D\_N*** та ***Num\_N\_D***, не залежно від того, в якому буде діапазоні основа системи числення (*n* ≤ 10 чи ні) програми можуть бути віднесені до одного виду (використовуються і числові і текстові дані), що дає значні дидактичні переваги при вивченні даної теми.

Вище розглядались програми на переведення чисел із десяткової системи числення у шістнадцяткову та навпаки, із шістнадцяткової системи в десяткову, а також із десяткової системи числення у двійкову і також навпаки, в десчткову. Аналогічно можна створити ще ряд програм для переведення чисел із десяткової системи числення у систему числення із основою *k* (*k* ‒ довільне натуральне число). Щоб краще відпрацювання навички на техніку програмування варто запропонувати для самостійного виконання учням завдання (див. табл.1).

Такі завдання не мають практичного застосування, зате вони дозволять довести до рівня автоматизації навички техніки програмування. Їх можна використовувати при роботі в групах і при створенні і застосуванні тестів, що завжди важливо при роботі з такими групами. Якщо для створення тестів при переведенні чисел між системами з основами 2, 8, 10 та 16 легко скористатись стандартним додатком Windows **КАЛЬКУЛЯТОР** в режимі **ПРОГРАМІСТ**, то створення тестів для чисел з іншими основами викличуть труднощі.

|  |  |
| --- | --- |
| *таблиця 1* | |
| ***тип переведення*** | |
| 10→3 | 3→10 |
| 10→4 | 4→10 |
| 10→5 | 5→10 |
| 10→6 | 6→10 |
| 10→7 | 7→10 |
| 10→8 | 8→10 |
| 10→9 | 9→10 |
| 10→11 | 11→10 |
| 10→12 | 12→10 |
| 10→13 | 13→10 |
| 10→14 | 14→10 |
| 10→15 | 15→10 |
| 10→17 | 17→10 |
| . . . | . . . |

Робота по складанню та тестуванню програм вищеописаного типу наштовхує ще на два типи задач технічного характеру:

1. засвоєння методу **низхідного проектування програм**, тобто, створення підпрограм (процедур і функцій користувача) та конструювання програми із їх використанням.
2. **переведення чисел між системами з основами числення *k* і *n*** (*k* та *m* ‒ довільні натуральні числа), наприклад, переведення натурального числа із трійкової системи числення в дев’яткову систему числення;

В основу методу низхідного проектування покладено поділ задачі на дрібніші підзадачі, які можна розв'язувати окремо. Алгоритми розв'язання підзадач розглядаються як підпрограми, іменами яких можна оперувати при розв'язанні загальної задачі. Отже, програма, що розв'язує загальну задачу, яка викликає підпрограми (декомпонування). Процес декомлозиції триває доти, доки не будуть отримані блоки, що є достатньо малими для їх безпосереднього кодування. При цьому керуючу програму проектують раніше, ніж реалізують її складові частини. Таким чином, програма ієрархічно структурується і розробляється шляхом послідовного уточнення на кожному рівні ієрархії. В основу цього процесу, принципи ієрархічності, абстрагування, специфікації інтерфейсів і модульності.

***Цікаві факти***:

HTML-кольори RGB (Red ‒ Червоний, Green ‒ Зелений, Blue ‒ Синій) записується як 3 двозначні числа hex від 0 до FF(25510) з попереднім знаком #, наприклад рожевий ‒ **#FF8080**, сірий ‒ **#808080**, чорний ‒ **#000000**. Цей запис стосується 24-бітного кольору, який приписують тому чи іншому графічному елементу документу HTML.

Традиційні китайські одиниці ваги базувалися на 16. Наприклад, один Чжин (斤) в старій системі становить шістнадцять Таеля. Суаньпань (китайська абака) може бути використаний для виконання шістнадцяткових обчислень.

***Запитання:***

1. Чи можуть бути способи переведення чисел з однієї позиційної системи в іншу “напряму”, без використання десяткової системи числення в якості “транзитної”?
2. Між якими позиційними системами числення існує “прямий” логічний звʹязок?

***Завдання:***

1. Написати програму переведення “напряму”, тобто без використання, як проміжної, десяткової системи числення двійкового числа в *n*-у систему числення (2 < *n* < 10) і без використання нечислових типів величин.
2. Написати програму переведення n-го числа “напряму”, тобто без використання десяткової системи числення у якості “транзитної”.

**ПЕРЕТВОРЕННЯ ДВІЙКОВОГО ЧИСЛА**

**4**

**В ШІСТНАДЦЯДКОВУ СИСТЕМУ ЧИСЛЕННЯ**

**І ШІСТНАДЦЯДКОВОГО ЧИСЛА**

**У ДВІЙКОВУ СИСТЕМУ ЧИСЛЕННЯ**

Щоб продемонструвати положення а) та б), розглянемо задачу представлення натурального числа, записаного в двійковій системі числення у вигляді числа, записаного в шістнадцятковій системі, тобто реалізацію схеми 2→16, яку, очевидно, можна трансформувати в схему: 2→10→16, одержавши програму ***Num\_2\_D\_16***.

Для цього необхідно за правилами мови програмування ***Pascal*** представити програми ***Num\_2\_D*** та ***Num\_D\_16***у вигляді функцій користувача. Ось їх ній вигляд:

**Функція *Num\_2\_D*** *function Num\_2\_D(a2:string):real;*

*begin*

*n:=Length(a2);aD:=0;j:=0;*

*for i:=1 to n do begin*

*a:=Copy(a2,n-i+1,1);j2:=Power(2,j);*

*if a='0' then aD:= aD+0\*j2 else*

*if a='1' then aD:= aD+1\*j2 else*

*if a='2' then aD:= aD+2\*j2 else*

*if a='3' then aD:= aD+3\*j2 else*

*if a='4' then aD:= aD+4\*j2 else*

*if a='5' then aD:= aD+5\*j2 else*

*if a='6' then aD:= aD+6\*j2 else*

*if a='7' then aD:= aD+7\*j2 else*

*if a='8' then aD:= aD+7\*j2 else*

*if a='9' then aD:= aD+7\*j2;*

*j:=j+1 end;*

*Num\_2\_D:=aD*

*end;*

**Функція *Num\_2\_D*** *function Num\_D\_16(aD:integer):string;*

*begin*

*a16:='';*

*while aD>0 do begin*

*aM:=aD mod 16;aD:=aD div 16;*

*if aM=0 then a16:='0'+a16 else*

*if aM=1 then a16:='1'+a16 else*

*if aM=2 then a16:='2'+a16 else*

*if aM=3 then a16:='3'+a16 else*

*if aM=4 then a16:='4'+a16 else*

*if aM=5 then a16:='5'+a16 else*

*if aM=6 then a16:='6'+a16 else*

*if aM=7 then a16:='7'+a16 else*

*if aM=8 then a16:='8'+a16 else*

*if aM=9 then a16:='9'+a16 else*

*if aM=10 then a16:='A'+a16 else*

*if aM=11 then a16:='B'+a16 else*

*if aM=12 then a16:='C'+a16 else*

*if aM=13 then a16:='D'+a16 else*

*if aM=14 then a16:='E'+a16 else*

*if aM=15 then a16:='F'+a16 end;*

*Num\_D\_16:=a16*

*end;*

Тоді програма ***Num\_2\_D\_16*** матиме вигляд:

***program Num\_2\_D\_16;***

***uses Crt;***

***var i,j,j2,n,aM,aD:integer;j1:real;***

***a2,a,a16:string;***

*function Num\_2\_D(a2:string):integer;*

*begin*

*n:=Length(a2);aD:=0;j:=0;*

*for i:=1 to n do begin*

*a:=Copy(a2,n-i+1,1);*

*j1:=Power(2,j);j2:=Trunc(j1);*

*if a='0' then aD:= aD+0\*j2 else*

*if a='1' then aD:= aD+1\*j2 else*

*if a='2' then aD:= aD+2\*j2 else*

*if a='3' then aD:= aD+3\*j2 else*

*if a='4' then aD:= aD+4\*j2 else*

*if a='5' then aD:= aD+5\*j2 else*

*if a='6' then aD:= aD+6\*j2 else*

*if a='7' then aD:= aD+7\*j2 else*

*if a='8' then aD:= aD+7\*j2 else*

*if a='9' then aD:= aD+7\*j2;*

*j:=j+1 end;*

*Num\_2\_D:=aD*

*end;*

*function Num\_D\_16(aD:integer):string;*

*begin*

*a16:='';*

*while aD>0 do begin*

*aM:=aD mod 16;aD:=aD div 16;*

*if aM=0 then a16:='0'+a16 else*

*if aM=1 then a16:='1'+a16 else*

*if aM=2 then a16:='2'+a16 else*

*if aM=3 then a16:='3'+a16 else*

*if aM=4 then a16:='4'+a16 else*

*if aM=5 then a16:='5'+a16 else*

*if aM=6 then a16:='6'+a16 else*

*if aM=7 then a16:='7'+a16 else*

*if aM=8 then a16:='8'+a16 else*

*if aM=9 then a16:='9'+a16 else*

*if aM=10 then a16:='A'+a16 else*

*if aM=11 then a16:='B'+a16 else*

*if aM=12 then a16:='C'+a16 else*

*if aM=13 then a16:='D'+a16 else*

*if aM=14 then a16:='E'+a16 else*

*if aM=15 then a16:='F'+a16 end;*

*Num\_D\_16:=a16*

*end;*

***begin***

***Write('a2?');Read(a2);***

***aD:=Num\_2\_D(a2);***

***a16:=Num\_D\_16(aD);***

***WriteLn(a16)***

***end.***

Сама програма ***Num\_2\_D\_16*** виділена напівжирним курсивом, а її тіло рамкою. Код програми можна скопіювати у вікно системи програмування ***Pascal*** (програма написана для ***PascalABC***, як найбільш сучасної, доступної та зручної версії в умовах загальноосвітніх шкіл).

Тепер наведемо коди функцій ***Num\_16\_D*** та ***Num\_D\_2***, які дозволять створити програму ***Num\_16\_D\_2***, яка буде перетворювати шістнадцяткове число у двійкову систему числення:

***program Num\_16\_D\_2;***

***uses Crt;***

***var i,j,j2,n,aM,aD:integer;j1:real;***

***a2,a,a16:string;***

*function Num\_16\_D(a16:string):integer;*

*begin*

*n:=Length(a16);aD:=0;j:=0;*

*for i:=1 to n do begin*

*a:=Copy(a16,n-i+1,1);*

*j1:=Power(16,j);j2:=Trunc(j1);*

*if a='0' then aD:= aD+0\*j2 else*

*if a='1' then aD:= aD+1\*j2 else*

*if a='2' then aD:= aD+2\*j2 else*

*if a='3' then aD:= aD+3\*j2 else*

*if a='4' then aD:= aD+4\*j2 else*

*if a='5' then aD:= aD+5\*j2 else*

*if a='6' then aD:= aD+6\*j2 else*

*if a='7' then aD:= aD+7\*j2 else*

*if a='8' then aD:= aD+8\*j2 else*

*if a='9' then aD:= aD+9\*j2 else*

*if a='A' then aD:= aD+10\*j2 else*

*if a='B' then aD:= aD+11\*j2 else*

*if a='C' then aD:= aD+12\*j2 else*

*if a='D' then aD:= aD+13\*j2 else*

*if a='E' then aD:= aD+14\*j2 else*

*if a='F' then aD:= aD+15\*j2;*

*j:=j+1 end;*

*Num\_16\_D:=aD*

*end;*

*function Num\_D\_2(aD:integer):string;*

*begin*

*a2:='';*

*while aD>0 do begin*

*aM:=aD mod 2;aD:=aD div 2;*

*if aM=0 then a2:='0'+a2 else*

*if aM=1 then a2:='1'+a2 else*

*if aM=2 then a2:='2'+a2 else*

*if aM=3 then a2:='3'+a2 else*

*if aM=4 then a2:='4'+a2 else*

*if aM=5 then a2:='5'+a2 else*

*if aM=6 then a2:='6'+a2 else*

*if aM=7 then a2:='7'+a2 else*

*if aM=8 then a2:='8'+a2 else*

*if aM=9 then a2:='9'+a2 else*

*if aM=10 then a2:='A'+a2 else*

*if aM=11 then a2:='B'+a2 else*

*if aM=12 then a2:='C'+a2 else*

*if aM=13 then a2:='D'+a2 else*

*if aM=14 then a2:='E'+a2 else*

*if aM=15 then a2:='F'+a2 end;*

*Num\_D\_2:=a2*

*end;*

***begin***

***Write('a16?');Read(a16);***

***aD:=Num\_16\_D(a16);***

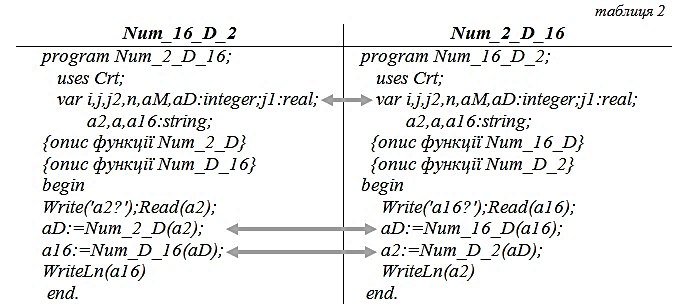
***a2:=Num\_D\_2(aD);***

***WriteLn(a2)***

***end.***

В програмах ***Num\_16\_D\_2*** та ***Num\_2\_D\_16*** для зручності сприйняття збережено імена всіх використаних змінних, крім того, також використано ті ж самі імена та типи формальних величин у функціях та у командах їх виклику в основних програмах. Це дасть можливість легше відслідковувати трасування програм із входом у підпрограми. Коди функцій у обох програмах для зручності порівняння взято також в рамки. У якості тренувальної вправи можна спробувати, маючи перед очима код програми ***Num\_2\_D\_16***, написати код та виконати програму ***Num\_16\_D\_2***.

В програмі ***Num\_16\_D\_2*** її основний код виділено напівжирним курсивом. Корисно цю частину порівняти із відповідною частиною програми ***Num\_2\_D\_16*** (для зручності нижче наведено в таблиці 2):



***Цікаві факти***:

Кожна шістнадцяткова цифра представляється чотирма бінарними цифрами (бітами), і основне застосування шістнадцяткового запису — це зручний запис двійкового коду. Одна шістнадцяткова цифра є ніблом, який є половиною з октету або байту (8 біт). Наприклад, значення байт лежить в діапазоні від 0 до 255 (в десяткових числах), але може бути більш зручно представити у вигляді двох шістнадцяткових цифр в діапазоні від 00 до FF.

Для переведення багатозначного двійкового числа у шістнадцяткову систему треба розбити його на тетради справа наліво та замінити кожну тетраду відповідною шістнадцятковою цифрою. Для переведення числа з шістнадцяткової системи у двійкову треба замінити кожну його цифру на відповідну тетраду з наведеної нижче таблиці переведення.

Наприклад:

0101101000112 = 0101 1010 0011 = 5A316 ;

0110 1000 0001 01112 = 681716;

1111 1111 1101 1111 1111 1011 1111 11112 = FFDFFBFF16;

111 1111 1111 1101 0111 1110 0000 0011 0111 1111 11112 = 0111 1111 1111 1101 0111 1110 0000 0011 0111 1111 11112 =7FFD7E037FF16.

Такий метод далі називатимемо ˮ**методом тетрад**ˮ.

***Запитання:***

1. Чи можна застосувати спосіб переведення чисел з двійкової системи у шістнадцяткову і навпаки для переведення чисел між двійковою та червірковою і між двійковою та восьмірковою системами числення?
2. Чи придатний подібний спосіб переведення між системами з іншими парами розрядних одиниць?

***Завдання:***

1. Написати програму переведення двійкового числа в шістнадцяткову систему числення з використанням тетрад.
2. Написати програму переведення щістнадцяткового числа в двійкову систему числення з використанням тетрад.

**СПОСОБИ ВВОДУ/ВИВОДУ ПАРАМЕТРІВ**

**5**

**У ПАСКАЛЬ-ПРОГРАМАХ**

При вивченні програмування не можна обійти способів вводу та виводу значень даних та результатів. На початкових етапах використовується виключно консольний спосіб, з допомогою стандартних процедур вводу та виводу. Необхідно виробити міцні навички виводу текстових, числових та формульних значень параметрів. Особливу увагу необхідно зосередити на форматованому виводі числових даних.

Окремим видом вводу/виводу є передача формальним параметрам фактичних параметрів при використанні підпрограм.

Буває зручним і доречним спосіб вводу параметрів, як констант. Зразок такого вводу є рядок ***const d : string[16] = '0123456789ABCDEF'*** у програмах ***N\_D\_M\_С*** та ***N\_D\_M\_F***, описаних нижче:

***const d : string[16] = '0123456789ABCDEF'***.

Після опрацювання названих способів вводу/виводу слід перейти від консольного до файлового вводу/виводу значень аргументів та результатів. Цю роботу необхідно розпочати із вивчення основних стандартних процедур роботи з файлами (див. таблицю 3).

|  |  |  |
| --- | --- | --- |
|  |  | *таблиця 3* |
| *№* | *формат* | *Виконання* |
| 1 | ***Assign(f:text,name:string)*** | Процедура, яка зв’язує файлову змінну *f* із файлом з іменем *name* |
| 2 | ***Reset(f:text)*** | Процедура, яка відкриває файл, раніше пов'язаний з файловою змінною *f* за допомогою процедури *Assign*. Файл повинен існувати на диску, в іншому випадку відбувається помилка часу виконання. Текстові файли відкриваються тільки на читання |
| 3 | ***Rewrite(f:text)*** | Процедура створює і відкриває новий файл, раніше пов'язаний з файловою змінною *f* за допомогою процедури *Assign*. Якщо файл з вказаним ім'ям вже існує, то замість нього створюється новий файл. Текстові файли відкриваються тільки на запис, типізовані файли - на читання і запис |
| 4 | ***Append(f:text)*** | Процедура відкриває текстовий файл на запис для додавання. Файловий покажчик встановлюється в кінець файлу |
| 5 | ***Close(f:text)*** | Процедура закриває відкритий для читання чи доповнення файл |
| 6 | ***Read(f,a,b,...:змінні простого типу, рядкові або вказівники)*** | Процедура зчитує значення з файлу *f* в змінні *а*, *b*... Якщо файл збірний, то типи змінних *а*, *b*... повинні збігатися з базовим типом файлу, а їх значення зчитуються з файлу в двійковому вигляді. Якщо файл текстовий, то значення змінних *а*, *b*... повинні зберігатися в файлі в текстовому вигляді |
| 7 | ***Readln(f,a,b,...,...:змінні простого типу, рядкові або вказівники)*** | Процедура зчитує значення з текстового файлу *f* в змінні *а*, *b* ..., після чого пропускає символи до кінця рядка. Виклик *ReadLn (f)* просто пропускає символи до кінця рядка |
| 8 | ***Write(f,a,b,...,...:змінні простого типу, рядкові або вказівники)*** | Процедура записує значення *а*, *b* ... в файл *f*. Якщо файл збірний, то типи значень *а*, *b*... повинні бути сумісними з базовим типом файлу. Якщо файл текстовий, то значення *а*, *b* ... виводяться в текстовому вигляді |
| 9 | ***Writeln(f,a,b,...,...:змінні простого типу, рядкові або вказівники)*** | Записує значення *а*, *b* ... в текстовий файл *f*, після чого записує в нього символ кінця рядка. Значення *а*, *b* ... записуються в файл у текстовому вигляді, . Виклик *riteLn (f)* просто записує в файл символ кінця рядка |
| 10 | ***EoLn(f)*** | Повертає *True*, якщо файловий вказівник стоїть в конці радку, і *False* в іншому випадку |
| 11 | ***Eof(f)*** | Повертає *True*, якщо файловий вказівник стоїть в конці файлу, і *False* в іншому випадку |

Слід додати, що наведені процедури доцільно використовувати в основній програмі, тому їх параметри також описуються в основній програмі, тобто є глобальними. Якщо слід ввод/вивод забезпечувати для структурованих типів величин, наприклад, масивів, а також коли дані у вхідному файлі записані в кількох рядках, то процедури ***Read/ReadLn*** та ***Write/WriteLn*** їх необхідно використовувати циклічно. Щоб це було зрозуміліше наведемо приклади програм із вводом масивів.

*Приклад 1:*  {програма створює масив випадкових чисел за

введеними кількостями рядків (*n*) та стовпців (*m*)}

*program* ***RW\_MAS****;*

*var fp:text;*

*j,i,m,n:integer;*

*a:array[1..100,1..100] of integer;*

*begin*

*Write('ввести n:');Read(n);*

*Write('ввести m:');Read(m);*

*Assign(fp,'f.txt');Rewrite(fp);*

*for i:=1 to n do begin*

*WriteLn(fp);*

*for j:=1 to m do begin*

*a[i,j]:=Random(12);*

*Write(fp,a[i,j],' ') end;end;*

*Close(fp);*

*Assign(fp,'f.txt');Reset(fp);*

*for i:=1 to n do begin*

*WriteLn;*

*for j:=1 to m do begin*

*Read(fp,a[i,j]);*

*Write(a[i,j],' ') end;end;*

*Close(fp);*

*end.*

Цей приклад ілюструє ввод циклами з параметром прямокутного масиву введених із клавіатури значень *n* та *m*.

Часто виникає потреба ввести із файлу кількість рядків та стовпців прямокутного масиву, тоді зручно буде використати цикли з післяумовою:

*Приклад 2. program* ***RESET\_MAS****;*

*var fp:text;i,j,m,n,s:integer;*

*a:array[1..5,1..5] of integer;*

*begin*

*Assign(fp,'f.dat');Reset(fp);*

*Read(fp,n); Read(fp,m); i:=1;j:=1;*

*repeat*

*repeat*

*Read(fp,a[i,j]);*

*i:=i+1*

*until* ***EoLn(fp)****;*

*j:=j+1;i:=1*

*until* ***Eof(fp)****;Close(fp);*

*s:=0;*

*for i:=1 to n do*

*for j:=1 to m do*

*s:=s+a[i,j];*

*WriteLn(s);*

*end.*

У цій програмі з допомогою одного звертання про відкриття файлу для читання вводяться кілька рядків, указаних в першому рядку, а вкладені цикли з післяумовою читають із текстового файлу числа, виділені в прикладі напівжирним шрифтом та рамкою. Звернемо увагу на умови циклів ***EoLn(fp)*** та ***EoF(fp)***. Це логічні (особливі!) функції, які можуть мати значення ***True*** або ***False***.

У текстовому файлі в кінці кожного рядка розміщений символ “кінець рядка”, а кожен файл закінчується символом “кінець файла” (див. малюнок 3). Умові Паскаль для відслідковування цих символів є логічні функції EoLn (End of Line ‒ кінець рядка), EoF(End of File ‒ кінець файла).



кінець файла

кінець рядка

*мал.4*

У програмі ***RESET\_MAS*** виконується повне читання усіх шести рядків, але знаходиться сума лише виділеного фрагмента. Щоб зрозуміти роботу цієї програми, потрібно виконати її покроково із відображенням проміжних значень у вікні відлагодження.

3 4

**1 2 3 4** 5

**1 2 3 4** 5

**1 2 3 4** 5

5 5 5 5 5

5 5 5 5 5

Після того, як у програмах ***Num\_2\_D\_16*** та ***Num\_16\_D\_2*** , виконано заміну консольного вводу/виводу на файловий, вони матимуть такий вигляд:

*program Num\_2\_D\_16;*

*var i,j,j2,n,aM,aD:integer;j1:real;*

*a2,a,a16:string;f:text;*

*procedure Inpt;*

*begin*

*Assign(f,'Num\_2\_D\_16.dat');Reset(f);*

*Read(f,a2);Close(f);*

*end;*

*procedure Outp;*

*begin*

*Assign(f,'Num\_2\_D\_16.res');ReWrite(f);*

*Write(f,a16);Close(f);*

*end;*

*function Num\_2\_D(a2:string):integer;*

*begin*

*n:=Length(a2);aD:=0;j:=0;*

*for i:=1 to n do begin*

*a:=Copy(a2,n-i+1,1);*

*j1:=Power(2,j);j2:=Trunc(j1);*

*if a='0' then aD:= aD+0\*j2 else*

*if a='1' then aD:= aD+1\*j2 else*

*if a='2' then aD:= aD+2\*j2 else*

*if a='3' then aD:= aD+3\*j2 else*

*if a='4' then aD:= aD+4\*j2 else*

*if a='5' then aD:= aD+5\*j2 else*

*if a='6' then aD:= aD+6\*j2 else*

*if a='7' then aD:= aD+7\*j2 else*

*if a='8' then aD:= aD+7\*j2 else*

*if a='9' then aD:= aD+7\*j2;*

*j:=j+1 end;*

*Num\_2\_D:=aD*

*end;*

*function Num\_D\_16(aD:integer):string;*

*begin*

*a16:='';*

*while aD>0 do begin*

*aM:=aD mod 16;aD:=aD div 16;*

*if aM=0 then a16:='0'+a16 else*

*if aM=1 then a16:='1'+a16 else*

*if aM=2 then a16:='2'+a16 else*

*if aM=3 then a16:='3'+a16 else*

*if aM=4 then a16:='4'+a16 else*

*if aM=5 then a16:='5'+a16 else*

*if aM=6 then a16:='6'+a16 else*

*if aM=7 then a16:='7'+a16 else*

*if aM=8 then a16:='8'+a16 else*

*if aM=9 then a16:='9'+a16 else*

*if aM=10 then a16:='A'+a16 else*

*if aM=11 then a16:='B'+a16 else*

*if aM=12 then a16:='C'+a16 else*

*if aM=13 then a16:='D'+a16 else*

*if aM=14 then a16:='E'+a16 else*

*if aM=15 then a16:='F'+a16 end;*

*Num\_D\_16:=a16*

*end;*

*begin*

*Inpt;{Write('a2?');Read(a2);}*

*aD:=Num\_2\_D(a2);*

*a16:=Num\_D\_16(aD);*

*Outp;{WriteLn(a16)}*

*end.*

*program Num\_16\_D\_2;*

*var i,j,j2,n,aM,aD:integer;j1:real;*

*a2,a,a16:string;f:text;*

*procedure Inpt;*

*begin*

*Assign(f,'Num\_16\_D\_2.dat');Reset(f);*

*Read(f,a16);Close(f);*

*end;*

*procedure Outp;*

*begin*

*Assign(f,'Num\_16\_D\_2.res');ReWrite(f);*

*Write(f,a2);Close(f);*

*end;*

*function Num\_16\_D(a16:string):integer;*

*begin*

*n:=Length(a16);aD:=0;j:=0;*

*for i:=1 to n do begin*

*a:=Copy(a16,n-i+1,1);*

*j1:=Power(16,j);j2:=Trunc(j1);*

*if a='0' then aD:= aD+0\*j2 else*

*if a='1' then aD:= aD+1\*j2 else*

*if a='2' then aD:= aD+2\*j2 else*

*if a='3' then aD:= aD+3\*j2 else*

*if a='4' then aD:= aD+4\*j2 else*

*if a='5' then aD:= aD+5\*j2 else*

*if a='6' then aD:= aD+6\*j2 else*

*if a='7' then aD:= aD+7\*j2 else*

*if a='8' then aD:= aD+8\*j2 else*

*if a='9' then aD:= aD+9\*j2 else*

*if a='A' then aD:= aD+10\*j2 else*

*if a='B' then aD:= aD+11\*j2 else*

*if a='C' then aD:= aD+12\*j2 else*

*if a='D' then aD:= aD+13\*j2 else*

*if a='E' then aD:= aD+14\*j2 else*

*if a='F' then aD:= aD+15\*j2;*

*j:=j+1 end;*

*Num\_16\_D:=aD*

*end;*

*function Num\_D\_2(aD:integer):string;*

*begin*

*a2:='';*

*while aD>0 do begin*

*aM:=aD mod 2;aD:=aD div 2;*

*if aM=0 then a2:='0'+a2 else*

*if aM=1 then a2:='1'+a2 else*

*if aM=2 then a2:='2'+a2 else*

*if aM=3 then a2:='3'+a2 else*

*if aM=4 then a2:='4'+a2 else*

*if aM=5 then a2:='5'+a2 else*

*if aM=6 then a2:='6'+a2 else*

*if aM=7 then a2:='7'+a2 else*

*if aM=8 then a2:='8'+a2 else*

*if aM=9 then a2:='9'+a2 else*

*if aM=10 then a2:='A'+a2 else*

*if aM=11 then a2:='B'+a2 else*

*if aM=12 then a2:='C'+a2 else*

*if aM=13 then a2:='D'+a2 else*

*if aM=14 then a2:='E'+a2 else*

*if aM=15 then a2:='F'+a2 end;*

*Num\_D\_2:=a2*

*end;*

*begin*

*Inpt;{Write('a16?');Read(a16);}*

*aD:=Num\_16\_D(a16);*

*a2:=Num\_D\_2(aD);*

*Outp;{WriteLn(a2)}*

*end.*

У наведених вище програмах ***Num\_2\_D\_16*** та ***Num\_16\_D\_2*** слід звернути увагу на на виділені рамкою та сірою штриховкою фрагменти. В рамках знаходяться процедури вводу та виводу з використанням текстових файлів. В обох програмах названо однаково процедури вводу ***Inpt*** (від *input*) та ***Outp*** (від *output*). Для максимального спрощення (адже описане адресоване учителям, які будуть проводити початкове ознайомлення із таким способом вводу/виводу) процедури не мають парвметрів. Команди виклику цих процедур в основних програмах мають також максимально спрощений вигляд (*Inpt*, *Outp*), а команди вводу/виводу через консоль для порівняння не вилучені, а закоментовані.

Якщо скористатись вхідними файлами ***Num\_2\_D\_16.dat*** та ***Num\_16\_D\_2.dat***, одержимо результуючі файли ***Num\_2\_D\_16.res***та ***Num\_16\_D\_2.res***. Цікаво, що вміст файлів ***Num\_2\_D\_16.dat*** та ***Num\_16\_D\_2.res*** однакові, тобто:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *таблиця 4* | | | | |
| ***Num10*** | ***Num\_2\_D\_16.dat*** | ***Num\_2\_D\_16.res*** | ***Num\_16\_D\_2.dat*** | ***Num\_16\_D\_2.res*** |
| 2017 | 11111100001 | 7E1 | 7E1 | 11111100001 |
| 4095 | 111111111111 | FFF | FFF | 111111111111 |

Ця обставина дозволить створювати тестуючі файли для таких програм, які можемо назвати *взаємно оберненими*, або і, взагалі, виконувати обернені перетворення натуральних чисел між системати числення. Проте, в наведеному матеріалі існує обмеження для вхідних даних типом *integer*, тобто максимальним числом для цього типу 2147483647.

***Цікаві факти***:

Файл в **Pascal ABC** представляє собою послідовність элементів одного типа, що зберігаються на диску. В **Pascal ABC** доступні два типи файлів - *типізовані* та *текстові*. Текстові файли зберігають символи, разділені на рядки символами #13 та #10 (кінець рядка та кінець файлу). Для опису текстового файлу використовуює стандартне ім’я типу text, а для описання типізованого файлу - конструкція **file** **of** тип элементів:

**var**

f1: **file** **of** real;

f2: text;

Файлові процедури та функції описуються в пункті “Процедури та функції для роботи з файлами”.

При роботі з типізованими файлами, наприклад, запис у файл здійснюється з допомогою процедури ***WRITE***. При виконанні її виклику *Write( f, вираз-типу-компонентів-файлая)* обчислюється значення виразу та присвоюється доступному елементу файла, після чого вказівник доступного елемента зсувається на 1 елемент. Наприклад:

***program****...*

***var****f :****file of****integer; x : integer*

***begin***

*...*

*ReWrite(f); x:=2;*

*Write(f, 1); Write(f, x);*

*...*

***end****.*

***Запитання:***

1. Чи можна прочитати із текстового файлу числові дані, записані в рядок без пропусків?
2. Яка різниця у виконанні процедур *Read*(*f*, ім’я файлу) та *ReadLn*(*f*, ім’я файлу)?
3. Яка різниця у виконанні процедур *ReWrite* та *Append*?

***Завдання:***

1. Написати програму, яка читає з текстового файлу рядок чисел, записаних через пропуск.
2. Написати програму, яка читає набір чисел, записаних в стовпчик і визначає кількість таких чисел.
3. Написати програму, яка читає із файлу текст, що містить понад 255 символів.

**ПЕРЕВЕДЕННЯ ЧИСЕЛ**

**6**

**МІЖ ДОВІЛЬНИМИ СИСТЕМАМИ ЧИСЛЕННЯ**

Вище описано єдиний підхід до структур програм переведення натуральних чисел із однієї позиційної системи в іншу. Поряд із кількома дидактичними перевагами такого підходу, є суттєвий недолік, який полягає в тому, що за зразком потрібно складати для кожної пари вибраних систем числення свої програми переведення чисел. У той же час хотілося б мати єдину універсальну програму, яка при заданні основ числення *n* та *m* і числа в *n*-ковій системі видавала б на виході число в *m*-ковій системі числення. Далі наведемо такі програми. У них застосовано той самий підхід до кількості, типах та назвах величин, що дасть додаткові дидактичні зручності у співставленні цих програм.

***program N\_D\_M\_C;***

***uses Crt;***

***const d : string[16]='0123456789ABCDEF';***

***var c1, c2, m, i : integer;***

***t, s : string;***

*function D\_N(n,r:integer):string;*

*begin*

*s:='';*

*repeat*

*s:=d[(n mod r)+1]+s;*

*n:=n div r;*

*until n=0;*

*D\_N:=s;*

*end;*

*function N\_D(n:string;r:integer):integer;*

*begin*

*m:=0;*

*while n[1]='0' do*

*Delete(n,1,1);*

*for i:=1 to Length(n) do*

*m:=m\*r+pos(n[i],d)-1;*

*N\_D:=m;*

*end;*

***begin***

***WriteLn('вхідна система:'); ReadLn(c1);***

***WriteLn('число:');ReadLn(t);***

***WriteLn('вихідна система:'); ReadLn(c2);***

***WriteLn(D\_N (N\_D (t, c1), c2));*** *{1}*

***end.***

***program N\_D\_M\_F;***

***const d : string[16] = '0123456789ABCDEF';***

***var c, c1, c2, i, m : integer;***

***t, s, st : string; f:text;***

*procedure Inpt;*

*begin*

*Assign(f,'N\_D\_M1.dat');Reset(f);*

*ReadLn(f,c1);ReadLn(f,t);ReadLn(f,c2);*

*Close(f)*

*end;*

*function D\_N(n,k:integer):string;*

*begin*

*s:='';*

*repeat*

*s:=d[(n mod k)+1]+s;*

*n:=n div k;*

*until n=0;*

*D\_N:=s;*

*end;*

*function N\_D(n:string;k:integer):integer;*

*begin*

*m:=0;*

*while n[1]='0' do*

*Delete(n,1,1);*

*for i:=1 to Length(n) do*

*m:=m\*k+pos(n[i],d)-1;*

*N\_D:=m;*

*end;*

*procedure Outp;*

*begin*

*Assign(f,'N\_D\_M1.res');ReWrite(f);*

*WriteLn(f, D\_N (N\_D (t, c1), c2)); {1}*

*Close(f);*

*end;*

***begin***

***Inpt;***

***Outp***

***end.***

При перевагах, що полягає в універсальності програм ***N\_D\_M\_C*** та ***N\_D\_M\_F*** вони суттєво складніші за наведені раніше, тому не варто пропонувати їх учням, які не мають ще достатніх знань і навичок у програмуванні мовою Паскаль. Як показує практика, учням важко трасувати і аналізувати ці програми. Ми спеціально виділили позначкою {1} відповідні рядки, зокрема:

***WriteLn(f, D\_N (N\_D (t, c1), c2)).***

Тут одночасно викликаються дві функції *D\_N* та *N\_D*, причому остання вкладена в першу, що ускладнює аналіз фактичних параметрів та їх взаємозв’язки. Але ці програми дають можливість швидко створювати тести на переведення *n*-кових чисел у *m*-кові. Наведемо одержані результати:

|  |  |  |  |
| --- | --- | --- | --- |
| *таблиця 5* | | | |
| *вхідна*  *основа* | *число* | *вихідна основа* | *Число* |
| 10 | 1 | 7 | 1 |
| 10 | 1 | 12 | 1 |
| 10 | 1 | 15 | 1 |
| 10 | 1 | 16 | 1 |
| 10 | 2409 | 2 | 100101101001 |
| 10 | 2409 | 3 | 1022020 |
| 10 | 2409 | 4 | 211221 |
| 10 | 2409 | 5 | 34114 |
| 10 | 2409 | 6 | 15053 |
| 10 | 2409 | 7 | 10011 |
| 10 | 2409 | 8 | 4551 |
| 10 | 2409 | 9 | 3266 |
| 10 | 2409 | 11 | 18A0 |
| 10 | 2409 | 12 | 1489 |
| 10 | 2409 | 13 | 1134 |
| 10 | 2409 | 14 | C41 |
| 10 | 2409 | 15 | AA9 |
| 10 | 2409 | 16 | 969 |
| 10 | 2147483647 | 2 | 1111111111111111111111111111111 |
| 10 | 2147483647 | 3 | 12112122212110202101 |
| 10 | 2147483647 | 4 | 1333333333333333 |
| 10 | 2147483647 | 5 | 13344223434042 |
| 10 | 2147483647 | 6 | 553032005531 |
| 10 | 2147483647 | 7 | 104134211161 |
| 10 | 2147483647 | 8 | 17777777777 |
| 10 | 2147483647 | 9 | 5478773671 |
| 10 | 2147483647 | 11 | A02220281 |
| 10 | 2147483647 | 12 | 4BB2308A7 |
| 10 | 2147483647 | 13 | 282BA4AAA |
| 10 | 2147483647 | 14 | 1652CA931 |
| 10 | 2147483647 | 15 | C87E66B7 |
| 10 | 2147483647 | 16 | 7FFFFFFF |

З одержаної таблиці 5 можна складати тести для перевірки роботи програми ***N\_D\_M\_F***, наприклад:

|  |  |
| --- | --- |
| ***N\_D\_M.dat*** | ***N\_D\_M.res*** |
| 5  13344223434042  12 | 4BB2308A7 |

На закінчення теми переведення числа з однієї позиційної системи в іншузауважимо, що вхідні дані обмежені десятковими числами в діапазоні від 1 до 2147483647 (потужність цілочисельного типу *integer*).

***Цікаві факти***:

Файл в **Pascal ABC** представляє собою послідовність элементів одного типа, що зберігаються на диску. В **Pascal ABC** доступні два типи файлів - *типізовані* та *текстові*. Текстові файли зберігають символи, разділені на рядки символами #13 та #10 (кінець рядка та кінець файлу). Для опису текстового файлу використовуює стандартне ім’я типу *text*, а для описання типізованого файлу - конструкція ***file******of*** тип элементів:

***var***

*f1:* ***file******of*** *real;*

*f2: text;*

Файлові процедури та функції описуються в пункті “Процедури та функції для роботи з файлами”.

При роботі з типізованими файлами, наприклад, запис у файл здійснюється з допомогою процедури ***WRITE***. При виконанні її виклику *Write( f, вираз-типу-компонентів-файлая)* обчислюється значення виразу та присвоюється доступному елементу файла, після чого вказівник доступного елемента зсувається на 1 елемент. Наприклад:

***program****...*

***var****f :****file of****integer; x : integer*

***begin***

*...*

*ReWrite(f); x:=2;*

*Write(f, 1); Write(f, x);*

*...*

***end****.*

***Запитання:***

1. Чи можна однією процедурою *Read* читати змішані дані (наприклад, числові і рядкові)?
2. У таблиці 5 можна помітити цікавий факт: для десяткового числа 2147483647? Значення у двійковій, четвірковій, восьмірковій та шістнадцятковій системах числення записуються подібно, першою цифрою є одиниця, а решта цифр однакові, дорівнюють найбільшій цифрі в цій системічислення. Чи випадково та з чим це пов’язано?

***Завдання:***

1. (*задачу взято із колекції, запропонованих на тренувальному турі ІІІ етапу олімпіади з програмування в 2017 році*) Як відомо, в двійковій системі ній необхідно вміти виконувати різного роду операції зі степенями двійки. Пропонується гра, правили якої полягають в наступному. В рядку знаходяться числа – степені двійки. Гравець може обрати два довільних однакових числа, після чого ці числа зникають, а на полі з’являється інше число, що рівне сумі обраних чисел. Написати програму, що за початковим набором чисел на полі, знаходить найбільше число, що може з’явитися під час гри.

*Вхідні дані*:

У першому рядку записане одне число N (1 ≤ N ≤ 216) – кількість чисел. Другий рядок містить N цілих чисел Ai (1 ≤ Ai ≤ 230) – числа, що записані на полі на початку гри.

*Результати*:

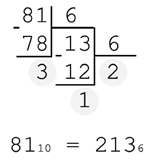
У єдиному рядку виведіть одне число – найбільше число, що може з’явитися на полі під час гри.

|  |  |
| --- | --- |
| **Input in c.in** | **Output in c.out** |
| 4  2 4 4 8 | 16 |
| 9  4 4 4 4 4 4 4 4 4 | 32 |

*Приклади:*

1. **Цікаве число** (*задачу взято із колекції, запропонованих на тренувальному турі ІІІ етапу олімпіади з програмування в 2017 році*) На факультативі з програмування почали вивчити системи числення. На першому уроці вчитель розповів про систему числення з основою два, дуже популярною в комп'ютерному світі. На другому уроці учні дізналися про систему числення з основою три. І так далі: на кожному наступному уроці дізнавались про нові системи числення, так що на *i*-му уроці була розглянута система числення з основою *i* +1.

*мал.5*



Щоб краще запам'ятати, на кожному уроці брали одне і те ж число *x* і записували його в зошит в останній вивченій системі числення.

Приклад перекладу числа 81 в систему числення з основою 6 (малюнок 5). Помічено, що у записаному числі *x* в новій системі числення всі цифри однакові. Якщо уявити, що таке відбувається вперше, і ні на якому з попередніх уроків число, не виходило таким цікавим. Під враженням забулось, яка система числення в цей день розглядалась на уроці. Потрібно написати програму, що знаходить систему числення з мінімальною основою, в якій це число має однакові цифри.

*Вхідні дані:*

Єдиний рядок вхідного файлу містить одне ціле число X (1 ≤ X ≤ 1012) - число записане в десятковій системі числення.

*Результати:*

Вихідний файл повинен містити одне ціле число B (2 ≤ B) - шукана система числення.

Пояснення до прикладів:

Перший приклад: "3" це "11" в системі числення з основою 2.

Другий приклад: "219" це "333" в системі числення з основою 8.

Третій приклад: "1009" це "11" в системі числення з основою 1008.

**ЗАСТОСУВАННЯ МЕТОДІВ БАГАТОРОЗРЯДНОЇ**

**7**

**АРИФМЕТИКИ В ПРОГРАМАХ ПЕРЕТВОРЕННЯ ЧИСЕЛ МІЖ ПОЗИЦІЙНИМИ СИСТЕМАМИ ЧИСЛЕННЯ**

**7.1 (*Стандартний спосіб*)** До цього моменту досліджувалось питання про переведення натуральних позиційних чисел в межах типу *integer*, але можуть становити інтерес розв’язування такої задачі для багаторозрядних натуральних чисел, наприклад, переведення даного натурального числа в 25-ковій системі числення у 35-кову систему числення. Для цього слід використовувати спеціальні методи опрацювання багаторозрядних чисел. Тут скористаємось стандартними процедурами і функціями виконання операцій над багаторозрядними десятковими числами на основі текстового представлення багаторозрядних чисел.

Нижче наведена програма ***D\_2\_LONG*** виконує переведення десяткових багатоцифрових чисел у двійкову систему числення. Видно, що вона досить громіздка за рахунок процедури ***LongMoDiv*** (визначення остачі від ділення та неповної частки при цілочисельному діленні багаторозрядних чисел) та функцій ***Comp*** (порівняння двох багатоцифрових чисел) та ***Sub*** (віднімання багатоцифрових чисел).

*program D\_2\_LONG;*

*var i:integer;*

*a,b,n,c,str1,str2,strMod,strDiv:string;*

*q:boolean;f:text;*

*function Comp(s1,s2:string):boolean;*

*var j,l1,l2:byte;st:string;*

*begin l1:=Length(s1);l2:=Length(s2);*

*if l1<l2 then for j:=l2-l1 downto 1 do s1:='0'+s1*

*else for j:=l1-l2 downto 1 do s2:='0'+s2;*

*if s1>=s2 then Comp:=true else Comp:=false*

*end;*

*function Sub(s1,s2:string):string;*

*var j,l1,l2,pm:byte;code,s\_1,s\_2,s:integer;*

*begin*

*l1:=Length(s1);l2:=Length(s2);pm:=0;*

*if s1=s2 then Sub:='0'*

*else begin*

*for j:=l1-l2 downto 1 do s2:='0'+s2;*

*for j:=l1 downto 1 do begin*

*Val(s1[j],s\_1,code);Val(s2[j],s\_2,code);*

*if s\_1>=s\_2+pm then begin s:=s\_1-(s\_2+pm);pm:=0 end*

*else begin s:=(s\_1+10)-(s\_2+pm);pm:=1 end;*

*s1[j]:=Chr(s+48) end;*

*while s1[1]='0' do Delete (s1,1,1);Sub:=s1 end*

*end;*

*procedure MoDiv(st1,st2:string;var stmod,stdiv:string);*

*var i,l,k, j:integer;st\_1:string;bZero: boolean;*

*begin*

*st\_1:='';stdiv:='';i:=1;l:=Length(st1);*

*repeat*

*k:=0;if st2='' then break;*

*while not Comp(st\_1,st2) do begin*

*st\_1:=st\_1+st1[i];i:=i+1;stdiv:=stdiv+'0';*

*if i>Length(st2) then break;bZero := true;*

*if Length(st\_1)>Length(st2) then begin*

*for j := 1 to Length(st\_1)-1 do*

*if st\_1[j]<>'0' then bZero := false;*

*if (st\_1[Length(st\_1)]<>str2[Length(str2)]) and bZero then*

*begin Delete(st\_1,Length(st\_1),1);break end end end;*

*Delete (stdiv,Length(stdiv),1);*

*while st\_1[1]='0' do begin*

*If st\_1 = '' then break; Delete (st\_1,1,1) end;*

*while Comp(st\_1,st2) do begin*

*st\_1:=Sub(st\_1,st2);k:=k+1 end;*

*stmod:=st\_1;*

*if st\_1 = '' then stmod := '0';*

*stdiv:=stdiv+Chr(k+48);*

*while Length(stdiv)>Length(st1) do*

*Delete(stdiv,Length(stdiv),1);*

*until i>l;*

*if (Length(stdiv)<>1) and (stdiv[1]='0') then*

*while stdiv[1]='0' do Delete(stdiv,1,1);*

*end;*

*procedure Inpt;*

*begin*

*Assign(f,'D2Lng5.dat');Reset(f);*

*Read(f,a);Close(f);*

*end;*

*procedure Proc;*

*begin*

*n:='2';q:=Comp(a,'0');*

*if a='0' then q:=False;*

*while q=True do begin*

*MoDiv(a,n,c,a);*

*for i:=0 to 9 do*

*if c=IntToStr(i) then b:=c+b;*

*q:=Comp(a,'0');*

*if a='0' then q:=False end;*

*end;*

*procedure Outp;*

*begin*

*Assign(f,'D2Lng5.res');*

*Rewrite(f);Write(f,b);Close(f);*

*end;*

*begin*

*Inpt;*

*Proc;*

Програма ***D\_2\_LONG*** дає такі результати (таблиця 6):

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | | *таблиця 6* |
| ***№*** | | ***D2Lng.dat*** | ***D2Lng.res*** |
| *1* | | 1000000000000000000 | 110111100000101101101011001110100111011001000000000000000000 |
| *2* | | 4000000000000000000 | 11011110000010110110101100111010011101100100000000000000000000 |
| *3* | | 4000000000000000001 | 11011110000010110110101100111010011101100100000000000000000001 |
| *4* | | 3106511852580896964 | 10101100011100100011000001001000100111101000000000000011000100 |
| *5* | | 123456789123456789123456789123456789123456789 | 101100010010011011011100101001111010001001110011010111111101111101010111011001001101000001111110001010100001011011010000100000001000101111100010101 |
| *6* | | 99999999999999999999999999999999999999999999999999999999999999999 | 111100110001011000100111000111000111111111000011100100001000101010001011111011110100011001001110001110010100010111101111011110100010010100110110000010011111111111111111111111111111111111111111111111111111111111111111 |
| *7* | | 99999999999999999999999999999999999999999999999999999999999999998 | 111100110001011000100111000111000111111111000011100100001000101010001011111011110100011001001110001110010100010111101111011110100010010100110110000010011111111111111111111111111111111111111111111111111111111111111111 |
| *8* | | 999999999999999999999999999999999999999999993 | 101100110101110110111111100000100001101011100100111100111000101111011101101000101000000000101100100010100111111111111111111111111111111111111111111001 |
| *9* | | 999999999999999999999999999999999999999999994 | 101100110101110110111111100000100001101011100100111100111000101111011101101000101000000000101100100010100111111111111111111111111111111111111111111010 |
| *10* | | 999999999999999999999999999999999999999999995 | 101100110101110110111111100000100001101011100100111100111000101111011101101000101000000000101100100010100111111111111111111111111111111111111111111011 |

Результати тестів перевірені. Вихідні результати в силу використання текстового способу опрацювання багатоціфрових чисел можуть містити не більше 255 знаків, що накладає свої обмеження і на вхідні дані, але і так видно, що програма дає можливість опрацьовувати астрономічні дані. Наприклад, п’ятий тест на виході дає 147 цифрове число, а вхідним 45-и цифрове число. Абсолютно точно визначити розмір найбільшого десяткового числа, яке опрацьлвує програма ***D\_2\_LONG*** да громіздко, але тут покажемо, як зробити таку ˮприкидкуˮ. У шостому тесті взято максимально можливе десяткове шестидесятип’яти цифрове число (1065-1). Програма видає відповідне двійкове число, але велика кількість одиниць у кінці ставить під сумнів правильність переводу. Щоб підтвердити таке припущення, можна взяти десяткове число 1065-2 (Теж 65 дев’яток, але замінено останню на цифру 8). Факт, що видає програма двійкове число, яке співпадає з попереднім результатом із шостого тесту, переконує, що чисел, близьких до 1065-1 програма ***D\_2\_LONG*** опрацьовувати не може. Очевидно, слід зменшувати кількість дев’яток у вхідному тесті. Так можна визначити гарантовано кількість цифр *n* вхідного числа, але результат у такому випадку знаходитиметься в діапазоні (10*n*-1-1; 10*n*-1). Якщо пошуки продовжити, то можна звузити діапазон достовірності (див. тести 8-9 в таблиці 6). При значеннях десяткового числа 1044+3 та 1044+4 двійкові коди останньому розряді різні, це значить двійкові коди десяткових чисел 1044+3 та 1044+4 достовірні. Зробимо “контрольний постріл”. В десятому рядку візьмемо десяткове число 1044+5 і одержимодвійковий код цього числа. З трьох останніх рядків видно, що в останніх чотирьох розрядах коди: 1001, 1010, 1011, що відповідно дорівнює десятковим числам 9, 10 та 11. Якщо при збільшенні десяткового числа на 1 відповідний двійковий код також збільшується на 1, то всі останні тести (8-10) не тільки достовірні, а й правильні. Чи можна стверджувати, що число 1044+5 для програми ***D\_2\_LONG***  граничне? Пропонується самостійно знайти відповідь на це питання.

З причини громіздкості подібних програм ми обмежились лише програмою ***D\_2\_LONG***, хоча, скориставшись продемонстрованим вище підходом цілком можливо було б навести коди програм ***2\_D\_LONG*** (переведення двійкового числа в десяткову систему), програми ***D\_16\_LONG*** і ***16\_D\_LONG***., що переводять шістнадцядкові багатоцифрові числа в десяткову систему і навпаки.

**7.2** (***Економний спосіб***) Але згадаємо наведені вище відомості про ˮметод тетрадˮ. Він дозволяє без використання громіздких і малозрозумілих багатьом методів опрацювання багаторозрядних чисел писати невеликі і схожі за ідеологією програмам з п. 3-4 програми переведення багаторозрядних двійкових, четвіркових, восьміркових, шістнадцяткових та інших чисел, розрядні одиниці яких є степенями двійки і програми для зворотного переведення. Цей метод дозволить напряму і без громіздких програмних кодів перетворювати між собою четвіркові, восьміркові, шістнадцяткові та інші числа без використання двійкової системи числення.

Для прикладу наведемо програму ***2\_H\_Long*** (з двійкової систему в шістнадцяткову).

***program B\_H\_Long;***

***var i,n,k,m:integer;***

***a,a2,aH:string;***

***f:text;***

*procedure Proc;*

*begin*

*n:=Length(a2);k:=n mod 4;*

*for i:=1 to 4-k do a2:='0'+a2;*

*n:=Length(a2);k:=n div 4;aH:='';m:=1;*

*for i:=1 to k do begin*

*a:=Copy(a2,m,4);m:=m+4;*

*if a='0000' then aH:=aH+'0' else*

*if a='0001' then aH:=aH+'1' else*

*if a='0010' then aH:=aH+'2' else*

*if a='0011' then aH:=aH+'3' else*

*if a='0100' then aH:=aH+'4' else*

*if a='0101' then aH:=aH+'5' else*

*if a='0110' then aH:=aH+'6' else*

*if a='0111' then aH:=aH+'7' else*

*if a='1000' then aH:=aH+'8' else*

*if a='1001' then aH:=aH+'9' else*

*if a='1010' then aH:=aH+'A' else*

*if a='1011' then aH:=aH+'B' else*

*if a='1100' then aH:=aH+'C' else*

*if a='1101' then aH:=aH+'D' else*

*if a='1110' then aH:=aH+'E' else*

*if a='1111' then aH:=aH+'F' end;*

*end;*

*procedure Inpt;*

*begin*

*Assign(f,'2\_H\_Long.dat');Reset(f);*

*Read(f,a2);Close(f)*

*end;*

*procedure Outp;*

*begin*

*Assign(f,'2\_H\_Long.res'); ReWrite(f);*

*Write(f,aH);Close(f)*

*end;*

***begin***

***Inpt;Proc;Outp end.***

***Тести:***

|  |  |
| --- | --- |
| *2\_H\_Long.dat* | *2\_H\_Long.res* |
| 11111100001 | 7E1 |
| 101100010110110001 | 2C5B1 |
| 1011000101101100011 | 58B63 |
| 111111111011111111111111111111111111111100111111111111111111101111111111111111111111111111000000000000011100111 | 7FDFFFFFFF9FFFFDFFFFFFE000E7 |

З останнього тесту видно, що програма ***2\_H\_Long*** опрацьовує багаторозрядні двійкові числа, переводячи їх в шістнадцяткову систему числення.

***Цікаві факти***:

Вище кілька разів зверталась увага на особливості систем числення з основами два, чотири, вісім та шістнадцять. Наприклад, в цікавих фактах, запитаннях та завданнях до п. 4 ішлось про представлення двійкового коду вшістнадцяткове число з допомогою тетрад. Очевидно, що за цим методом легко представляти і багаторозрядні числа, наприклад:

11111111101111111111111111111111111111110011111111111111111110111111111111111111111111111100000000000001110011111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111000011111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111100011101112 =

=FFBFFFFFFF3FFFFBFFFFFFC001CFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF87FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFC7716 (близько 480 двійкових цифр, понад 110 шістнадцяткових цифр).

Це дуже економний спосіб і в програмному коді і в швидкодії програми опрацьовувати дуже багатоцифрові числа у відповідних системах числення (порівняймо хоча б із програмою ***D\_2\_LONG***).

***Запитання:***

Для яких систем числення з основою s<130 можливо застосувати метод переведення двійкових чисел у ці системи?

***Завдання:***

1. Дано 1000-цифрове двійкове число. Визначити кількість цифр цього числа у 16-й системі числення.
2. Написати програму, що визначає кількість цифр 8-го числа, переведеного в 4-у систему числення.
3. Написати програму переведення багатоцифрового шістнадцяткового числа у двійкову систему числення економним способом (методом тетрад).

**8**

**ДЕЯКІ “СЕКРЕТИ” ДЛЯ ЮНИХ ПРОГРАМІСТІВ**

Матеріали пропонованого навчально-методичного посібника націлені на розкриття особливостей початкового етапу серйозного навчання програмуванню школярів, яке не можна уявити без змагань. Такими змаганнями традиційно є різні етапи Всеукраїнської олімпіади з програмування для учнів. Тому крім навичок програмування необхідно знати і майстерно застосовувати найпростіші “секрети” таких змагань. Як відомо бали “не пахнуть”, подібно висловленню про гроші. Крім досвіду, ерудиції та інтуіції треба вміти набрати якомога більше балів, що підвищить рейтинг юного програміста і, неминуче, його позитивну мотивацію.

Серед задач, які пропонуються, завжди є одна-дві легких, які умовно можна назвати “втішальними” а також дуже складні, інакше ‒ “непідйомними”. Отже потрібно вміти відсіяти “зерна” від “полови”. Для цього варто при ознайомленні із завданнями деякий, можливо значний, час виділити для дослідження умов задач та роботи над побудовою математичних моделей і опису та дослідження алгоритмів. Це допоможе спочатку всі зусилля присвятити задачам, які дозволять набрати можливий максимум балів. Решту часу цілком можна приділити важчим задачам.

Щоб досягти найбільшої кількості балів потрібно пильно дослідити умови задач, звернувши особливу увагу вхідним даним і результатам та наведеним прикладам тестів. Задачу варто самостійно перевірити на тестах. Але при їх відсутності потрібно навчитись самостійно складати найбільш вірогідні тести. Характер тестів випливає із нюансів, зазначених в початкових умовах, накладених на вхідні дані та результати). Тести можна поділити на ті, що легко і швидко опрацьовуються (схожі на наведені в прикладах і призначені для перевірки можливості компіляції програми), основні та екстремальні. Основні оцінюються, як правило, середностатистичними балами і призначені перевірити програму на стандартних ситуаціях. Екстремальні змушують виявити здатність швидкодію програми та можливі виходи за діапазони вхідних даних.

Так звані легкі тести дозволять набрати не більше кількох балів або до двох-п’яти процентів від загальної кількості. На середніх тестах вдається додати 20-40% балів. А екстремальні тести (їх може бути всього одиниці) часто “важать” від 50 до 80 процентів балів.

Проходження легких тестів можна передбачити практично завжди, при цьому набрати кілька процентів можливих балів. Для недосвідченого програміста це також вихід. Якщо пройти всі легкі тести всіх запропонованих задач, можна одержати цілком втішний результат. Наприклад, дано п’ять задач із загальною кількістю 400 500 балів. 5% від них становить 20-25 балів, що становить до половини балів задачі звичайної складності. Якщо “взяти” ще середні тести однієї із задач, це в сумі може становити до 10-15% загальної кількості балів тобто 40-60 балів. І ці міркування зовсім не з категорії “дурень думкою багатіє”.

Оптимально було б вчасно і правильно визначити одну-дві реально посильні задачі, набравши на яких 20-30% балів, тоді з вище врахованими навіть 5% додаткових легких балів вони дозволили б ввійти до категорії учасників, недалеких від призового місця (часто дипломи І-ІІІ ступеня присуджуються до половини учасникам).

На такі результати можна сподіватись навіть на ІІІ-у (обласному) етапі. На жаль ‒ це потолок для учасників олімпіади, які оволоділи поки-що стандартними, тобто технічними уміннями та навичками програмування. Більшого сподіватись можна тільки при міцному засвоєнні спеціальних прийомів програмування. Ось деякі з них:

1. Елементи дискретної математики;
2. Сортування та пошук;
3. Елементи теорії графів;
4. Оптимальні способи перебору;
5. Елементи обчислювальної геометрії;
6. Елементи теорії графів;
7. Методи опрацювання багатоцифрових чисел.

Ці методи та прийоми названі умовно, адже деякі з них мають спільні теоретичні засади та завдання. Напиклад, дискретна математика стосується не тільки комбінаторних задач, а й елементів теорії графів, суміжна із методами сортування та пошуку, способів перебору. Комбінаторні задачі поділяються на задачі підрахунку та задачі генерування множин елементів. Є також багато видів сортування та пошуку. Відомо багато методів оптимізації, виділимо окремо, наприклад, метод динамічного програмування. Щодо задач так званої “довгої арифметики”, то вони можуть базуватись на табличному, рядковому чи іншому принципі представлення багатоціфрових чисел.

І все це тільки елементи звичайної ерудиції, які, правда, часто виходять за межі елементарної математики. Окрім них варто назвати так звані евристичні алгоритми чи логічні методи опрацювання даних. При застосуванні названих методів структурного програмування потрібні, по-перше, неабиякі знання та практика в галузі програмування, по-друге, те, чого не додасть ні теорія, ні практика, тобто справжній програмістський талант.

На перший погляд, стандартна задача може виявитись надзвичайно складною, наприклад, якщо змінити діапазон даних,

Наведемо у якості ілюстрацій

**ВИСНОВКИ**

**9**

У запропонованому матеріалі розглянуто цілісний підхід до вводу в методи структурного програмування, яке в шкільних програмах рівня стандарту та академічного не розглядається, тому цей підхід на основі багаторічного досвіду може бути своєрідним містком між шкільним програмуванням та таким, що вивчається на перших курсах у вищих навчальних закладів. Цей матеріал і сам підхід будуть корисними для учителів, які намагаються зрозуміти з чого розпочати підготовку учнів до шкільних олімпіад з програмування.

У звʹязку з цим вибрано і з найменшими деталями розглянуто глобальну задачу опрацювання чисел в різних позиційних системах з екскурсами в методи вводу/виводу, зокрема з використанням зовнішніх текстових файлів, а також в методи розвʹязування задач багаторозрядної арифметики.

Цей матеріал буде слушним додатком до навчально-методичного доробку автора, що вже у різні роки публікувався в, зокрема, фаховій періодичній пресі (див. літературу). Таким чином, всім бажаючим можна буде достатньо ґрунтовно розібратись в елементах структурного програмування мовою Паскаль, в методах опрацювання текстових (рядкових) величин, методах перебору і зокрема методі динамічного програмування, розібрати окремі задачі опрацювання графів та цілий ряд задач на програмування олімпіадного характеру.

Програмування органічно повʹязане з математикою, особливо з дискретною математикою. Воно виконує завдання формування математичних моделей задач та пошуку їх реалізації, допомагає розвивати до високого рівня логічне та алгоритмічне мислення.

У пропонованій роботі практично всі ідеї, дослідження та програмні розробки авторські, в той же час використано деякі загальновідомі та доступні матеріали з інтернету, що не порушує нічиїх авторських прав, а також дещо із літератури інших авторів, на які є посилання в авторських статтях, наведених у літературі. У зв’язку з цим у списку літератури наведено тільки власні друковані джерела. У роботі для економії часу не наведено прямих посилань на конкретні джерела та відповідні сторінки із них. Але тим, хто прочитає пропонований матеріал “діагональний” або більш детальний перегляд вказаних джерел обов’язково вкаже місця на логічно пов’язані фрагменти, адже весь кількадесятирічний педагогічний досвід автора можна розглядади, як єдине ціле, підпорядковане єдиним принципам.

В кінці кожного пункту є рубрика “Цікаві факти”, кілька запитань та завдань. Радимо обов’язково обміркувати їх, це додасть значні відсотки до розглянутого в кожному пункті.

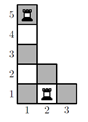
І все ж: з чого почати програмування, як поставлено питанням у назві? Дуже просто: **починати слід від моделі, через алгоритм та програму, від простого, до складного, від конкретної локальної задачі до її глобалізації, “накручуючи” все складніші та ефективніші прийоми, ідучи щляхом практики програмування**.

Про математичну постановку задачі, опис математичної моделі та опис і дослідження алгоритму ще не згадувалось, тому на закінчення цим питанням і присвятимо увагу. Для прикладу візьмемо дуже просту задачу, яка може здатись не такою тривіальною.

*Задача* ***Дивні шахи*** (*тренувальний тур олімпіади до ІІІ етапу олімпіади з програмування в 2017 р.*) Степан нещодавно придумав свою версію шахів, в якій гра відбувається на дошці, що має іншу форму. Його дошка складається з *N* стовпців, *i*-й з яких містить *Ai* клітин. Нижні клітини всіх стовпців утворюють один горизонтальний ряд, причому довжини стовпців впорядковані зліва направо по незростанню. На малюнку нижче наведений приклад дошки, в якій три стовпчика, містять 5, 2 і 1 клітинку, відповідно. Сьогодні Степана зацікавило питання: як розставити мінімальну кількість тур на його дошці так, щоб кожну клітинку поля била хоча б одна тура. Тура б’є ті клітини, які розташовані з нею на одній вертикалі або одній горизонталі. Допоможіть Cтепану розставити на його дошці мінімальне число тур потрібним чином.

*Вхідні дані*: Перший рядок вхідного файлу містить ціле число *N* (1 ≤ *N* ≤ 1000) ‒ кількість стовпців дошки. Наступний рядок містить *N* чисел *A1, A2, . . . , AN ‒* кількість клітинок в стовпцях (1 ≤ *Ai* ≤ 1000, *A1* ≥ *A2* ≥ . . . ≥ *AN*).

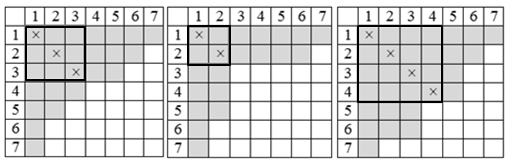
*Результати*: У першому рядку виведіть число *K* - мінімальна кількість тур, яку можна розставити на дошці так, щоб кожна клітинка дошки била хоча б одна тура. Наступні *K* рядків повинні містити опис позицій тур, по одній на кожному рядку. Позиція тури задається двома числами: номером стовпця, в якому стоїть тура, і номером клітинки в стовпці. Стовпці нумеруються, починаючи з 1, зліва направо, клітини в стовпцях нумеруються знизу вгору, також починаючи з 1. Якщо підходящих розстановок декілька, можна вивести будь-яку.



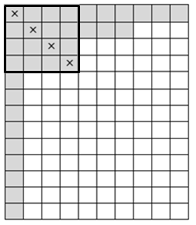
мал. 6

*Приклад***:**

|  |  |
| --- | --- |
| *Input in a.in* | *Output in a.out* |
| 3  5 2 1 | 2  1 5  2 1 |



*мал. 7*



*мал. 8*



*мал. 9*

***ЛІТЕРАТУРА***:

1. В.С. Остапець, Комп’ютерне моделювання й основи алгоритмізації та програмування у формі експрес-курсу, “Інформатика“, № 10, 2009 р., с. 9-16, № 11, 2009 р., с. 3-13, № 13, 2009 р., с. 6-16, № 14, 2009 р., с. 9-18.
2. В.С. Остапець, Шкільні олімпіади з інформатики, “Світло”, № 1, 2001 р., с. 9-12.
3. В.С. Остапець, Використання багаторозрядної арифметики, “Інформатика“, № 17, 2005 р. с. 17-23.
4. В.С. Остапець, Від задачі до задачі, “Інформатика“, № 33-34, 2005 р., с. 34-41.
5. В.С. Остапець, Метод динамічного програмування у школі, “Інформатика“, №9, 2008 р., с. 3-17.
6. В.С. Остапець, Позакласна робота з програмування у школі, навчально-методичний посібник, “Ризографіка“, Бориспіль, 2007 р. с. 3-115.
7. В.С. Остапець, Сучасний бінарний урок як найвищий ступінь міжпредметних зв’язків, “Інформатика“, № 43, 2009 р., с. 11-16.
8. В.С. Остапець, Про планування курсу інформатики в 5-6 класах за новими програмами, “Комп’ютер у школі та сім’ї”, № 7, 2013 р.,
9. В.С. Остапець, Письмовий опис алгоритмів як невідʹємна складова розвʹязку задачі сіз програмування у школі, “Компʹютер в школі та сімʹї”, №7, 2003 р., с. 31-33.
10. В.С. Остапець, Комплексні електронні дидактичні засоби в системі ІТН, методичний посібник для вчителів, видано за сприяння Бориспільської Районної Ради та благодійного фонду “Борисове поле”, с. 3-70
11. Сучасний урок інформатики, збірник (за матеріалами роботи слухачів ОШППД Остапця В.С.), Біла церква 2012 р., с. 3-147.

**ДОДАТКИ**

**10**

Зразки програм мовою pascal для переведення чисел з однієї позиційної системи числення в іншу. Із заголовка програми легко зрозуміти з якої системи числення в яку буде переведене число, наприклад: *Num\_2\_D* - програма переводить двійкове число у десяткове, *D2N* – із застосуванням процедур та функцій та тспособів вводу/виводу, зокрема і з використанням текстових файлів.

Додатки призначені на допомогу вчителям інформатики для створення індивідуальних завдань та тестів для учнів при підготовці до олімпіад та поглибленого вивчення розділу “Алгоритмізація і програмування”, написання робіт МАН.

При бажанні і мінімуму креативу можна давати учням завдання до тем на математичну модель задачі з програмування, створення агналогічних програм мовами FreePascal, С++, як консольні додатки так і візуальні проекти, а також поняття макросу та тести засобами Microsoft Office з допомогою Visual Basic.

*program Num\_2\_D;uses Crt;{двійкова - десяткова}*

*var a2,a:string;i,n:integer;aD,j,j2:real;*

*begin*

*Write('a2?');Read(a2);*

*n:=Length(a2);aD:=0;j:=0;*

*for i:=1 to n do begin*

*a:=Copy(a2,n-i+1,1);j2:=Power(2,j);*

*if a='0' then aD:= aD+0\*j2 else*

*if a='1' then aD:= aD+1\*j2 else*

*if a='2' then aD:= aD+2\*j2 else*

*if a='3' then aD:= aD+3\*j2 else*

*if a='4' then aD:= aD+4\*j2 else*

*if a='5' then aD:= aD+5\*j2 else*

*if a='6' then aD:= aD+6\*j2 else*

*if a='7' then aD:= aD+7\*j2 else*

*if a='8' then aD:= aD+8\*j2 else*

*if a='9' then aD:= aD+9\*j2;*

*j:=j+1 end;*

*WriteLn(aD);*

*end*.

*program Num\_2\_D\_Case;uses Crt;{двійкова-десяткова, з використанням оператора case}*

*var a2,a:string;i,n:integer;aD,j,j2:real;b:char;*

*begin*

*Write('a2?');Read(a2);*

*n:=Length(a2);aD:=0;j:=0;*

*for i:=1 to n do begin*

*a:=Copy(a2,n-i+1,1);b:=a[1];j2:=Power(2,j);*

*case b of*

*'0':aD:= aD+0\*j2;*

*'1':aD:= aD+1\*j2;*

*'2':aD:= aD+2\*j2;*

*'3':aD:= aD+3\*j2;*

*'4':aD:= aD+4\*j2;*

*'5':aD:= aD+5\*j2;*

*'6':aD:= aD+6\*j2;*

*'7':aD:= aD+7\*j2;*

*'8':aD:= aD+8\*j2;*

*'9':aD:= aD+9\*j2;*

*end;*

*j:=j+1 end;*

*WriteLn(aD);*

*end.*

*program D2N;*

*uses Crt;*

*var aD,i,c,n:integer;*

*a2:string;f:text;*

*{procedure Inpt;*

*begin*

*Assign(f,'D2N1.dat');Reset(f);*

*Read(f,aD);Close(f);*

*end;}*

*{procedure Proc;*

*begin*

*n:=2;*

*while aD>0 do begin*

*c:=aD mod n;aD:=aD div n;*

*for i:=0 to 9 do*

*if c=i then a2:=IntToStr(c)+a2 end;*

*end;}*

*{procedure Outp;*

*begin*

*Assign(f,'D2N1.res');Rewrite(f);*

*Write(f,a2);Close(f);*

*end;}*

*begin*

*Write('aD?');Read(aD);*

*{Inpt;}*

*n:=2;*

*while aD>0 do begin*

*c:=aD mod n;aD:=aD div n;*

*for i:=0 to 9 do*

*if c=i then a2:=IntToStr(c)+a2 end;*

*{Proc;}*

*{Outp;}*

*WriteLn(a2);*

*end.*

*program Num\_8\_D;uses Crt;*

*var a8,a:string;i,n:integer;aD,j,j8:real;*

*begin*

*Write('a8?');Read(a8);*

*n:=Length(a8);aD:=0;j:=0;*

*for i:=1 to n do begin*

*a:=Copy(a8,n-i+1,1);j8:=Power(8,j);*

*if a='0' then aD:= aD+0\*j8 else*

*if a='1' then aD:= aD+1\*j8 else*

*if a='2' then aD:= aD+2\*j8 else*

*if a='3' then aD:= aD+3\*j8 else*

*if a='4' then aD:= aD+4\*j8 else*

*if a='5' then aD:= aD+5\*j8 else*

*if a='6' then aD:= aD+6\*j8 else*

*if a='7' then aD:= aD+7\*j8;*

*j:=j+1 end;*

*WriteLn(aD);*

*end.*

*program Num\_9\_D;uses Crt;*

*var a9,a:string;i,n:integer;aD,j,j9:real;*

*begin*

*Write('a9?');Read(a9);*

*n:=Length(a9);aD:=0;j:=0;*

*for i:=1 to n do begin*

*a:=Copy(a9,n-i+1,1);j9:=Power(9,j);*

*if a='0' then aD:= aD+0\*j9 else*

*if a='1' then aD:= aD+1\*j9 else*

*if a='2' then aD:= aD+2\*j9 else*

*if a='3' then aD:= aD+3\*j9 else*

*if a='4' then aD:= aD+4\*j9 else*

*if a='5' then aD:= aD+5\*j9 else*

*if a='6' then aD:= aD+6\*j9 else*

*if a='7' then aD:= aD+7\*j9 else*

*if a='8' then aD:= aD+8\*j9;*

*j:=j+1 end;*

*WriteLn(aD);*

*end.*

*program Num\_16\_D;uses Crt;*

*var a16,a:string;i,n:integer;aD,j,j16:real;*

*begin*

*Write('a16?');Read(a16);*

*n:=Length(a16);aD:=0;j:=0;*

*for i:=1 to n do begin*

*a:=Copy(a16,n-i+1,1);j16:=Power(16,j);*

*if a='0' then aD:= aD+0\*j16 else*

*if a='1' then aD:= aD+1\*j16 else*

*if a='2' then aD:= aD+2\*j16 else*

*if a='3' then aD:= aD+3\*j16 else*

*if a='4' then aD:= aD+4\*j16 else*

*if a='5' then aD:= aD+5\*j16 else*

*if a='6' then aD:= aD+6\*j16 else*

*if a='7' then aD:= aD+7\*j16 else*

*if a='8' then aD:= aD+8\*j16 else*

*if a='9' then aD:= aD+9\*j16 else*

*if a='A' then aD:=aD+10\*j16 else*

*if a='B' then aD:=aD+11\*j16 else*

*if a='C' then aD:=aD+12\*j16 else*

*if a='D' then aD:=aD+13\*j16 else*

*if a='E' then aD:=aD+14\*j16 else*

*if a='F' then aD:=aD+15\*j16;*

*j:=j+1 end;*

*WriteLn(aD);*

*end.*

Як видно із заголовка програми, вона мусить переводити число із 17-річної системи у десяткову. Перевіримо:

100017 = 1\*173 + 0\*172 + 0\*171 + 0\*170 =173 = 491310;

101017 = 1\*173 + 0\*172 + 1\*171 + 0\*170 =173 = 493010;

101217 = 1\*173 + 0\*172 + 1\*171 + 1\*170 =173 = 493110;

*program Num\_17\_D;uses Crt;*

*var a17,a:string;i,n:integer;aD,j,j17:real;*

*begin*

*Write('a17?');Read(a17);*

*n:=Length(a17);aD:=0;j:=0;*

*for i:=1 to n do begin*

*a:=Copy(a17,n-i+1,1);j17:=Power(17,j);*

*if a='0' then aD:= aD+0\*j17 else*

*if a='1' then aD:= aD+1\*j17 else*

*if a='2' then aD:= aD+2\*j17 else*

*if a='3' then aD:= aD+3\*j17 else*

*if a='4' then aD:= aD+4\*j17 else*

*if a='5' then aD:= aD+5\*j17 else*

*if a='6' then aD:= aD+6\*j17 else*

*if a='7' then aD:= aD+7\*j17 else*

*if a='8' then aD:= aD+7\*j17 else*

*if a='9' then aD:= aD+7\*j17;*

*j:=j+1 end;*

*WriteLn(aD);*

*end.*

*program Num\_D\_8;uses Crt;*

*var a8:string;aM,aD:integer;*

*begin*

*Write('десяткове число?');Read(aD);a8:='';*

*while aD>0 do begin*

*aM:=aD mod 8;aD:=aD div 8;*

*if aM=0 then a8:='0'+a8 else*

*if aM=1 then a8:='1'+a8 else*

*if aM=2 then a8:='2'+a8 else*

*if aM=3 then a8:='3'+a8 else*

*if aM=4 then a8:='4'+a8 else*

*if aM=5 then a8:='5'+a8 else*

*if aM=6 then a8:='6'+a8 else*

*if aM=7 then a8:='7'+a8 end;*

*WriteLn(a8);*

*end.*

10010 = 1219 = 92 + 2\*91 + 90 = 81 + 18 + 1 = 10010 ;

11010 = 1329 = 81 + 27 + 2 = 10010.

*program Num\_D\_9;uses Crt;*

*var a9:string;aM,aD:integer;*

*begin*

*Write('десяткове число?');Read(aD);a9:='';*

*while aD>0 do begin*

*aM:=aD mod 9;aD:=aD div 9;*

*if aM=0 then a9:='0'+a9 else*

*if aM=1 then a9:='1'+a9 else*

*if aM=2 then a9:='2'+a9 else*

*if aM=3 then a9:='3'+a9 else*

*if aM=4 then a9:='4'+a9 else*

*if aM=5 then a9:='5'+a9 else*

*if aM=6 then a9:='6'+a9 else*

*if aM=7 then a9:='7'+a9 else*

*if aM=8 then a9:='8'+a9 end;*

*WriteLn(a9);*

*end.*

*program Num\_D\_11;uses Crt;*

*var a11:string;aM,aD:integer;*

*begin*

*Write('десяткове число?');Read(aD);a11:='';*

*while aD>0 do begin*

*aM:=aD mod 11;aD:=aD div 11;*

*if aM=0 then a11:='0'+a11 else*

*if aM=1 then a11:='1'+a11 else*

*if aM=2 then a11:='2'+a11 else*

*if aM=3 then a11:='3'+a11 else*

*if aM=4 then a11:='4'+a11 else*

*if aM=5 then a11:='5'+a11 else*

*if aM=6 then a11:='6'+a11 else*

*if aM=7 then a11:='7'+a11 else*

*if aM=8 then a11:='8'+a11 else*

*if aM=9 then a11:='9'+a11 else*

*if aM=10 then a11:='10'+a11 else*

*if aM=11 then a11:='11'+a11 end;*

*WriteLn(a11);*

*end.*

*program Num\_D\_16;uses Crt;*

*var a16:string;aM,aD:integer;*

*begin*

*Write('десяткове число?');Read(aD);a16:='';*

*while aD>0 do begin*

*aM:=aD mod 16;aD:=aD div 16;*

*if aM=0 then a16:='0'+a16 else*

*if aM=1 then a16:='1'+a16 else*

*if aM=2 then a16:='2'+a16 else*

*if aM=3 then a16:='3'+a16 else*

*if aM=4 then a16:='4'+a16 else*

*if aM=5 then a16:='5'+a16 else*

*if aM=6 then a16:='6'+a16 else*

*if aM=7 then a16:='7'+a16 else*

*if aM=8 then a16:='8'+a16 else*

*if aM=9 then a16:='9'+a16 else*

*if aM=10 then a16:='A'+a16 else*

*if aM=11 then a16:='B'+a16 else*

*if aM=12 then a16:='C'+a16 else*

*if aM=13 then a16:='D'+a16 else*

*if aM=14 then a16:='E'+a16 else*

*if aM=15 then a16:='F'+a16 end;*

*WriteLn(a16);*

*end.*

Ця програма переводить двійкове число у шіснадцяткову систему числення, використовуючи *Num\_2\_D* (переведення з двійкової системи числення у десяткову систему числення) та *Num\_D\_16* (переведення з десяткової системи числення у шіснадцяткову систему числення).

*program Num\_2\_D\_16;*

*uses Crt;*

*var i,j,j2,n,aM,aD:integer;j1:real;*

*a2,a,a16:string;*

*function Num\_2\_D(a2:string):integer;*

*begin*

*n:=Length(a2);aD:=0;j:=0;*

*for i:=1 to n do begin*

*a:=Copy(a2,n-i+1,1);*

*j1:=Power(2,j);j2:=Trunc(j1);*

*if a='0' then aD:= aD+0\*j2 else*

*if a='1' then aD:= aD+1\*j2 else*

*if a='2' then aD:= aD+2\*j2 else*

*if a='3' then aD:= aD+3\*j2 else*

*if a='4' then aD:= aD+4\*j2 else*

*if a='5' then aD:= aD+5\*j2 else*

*if a='6' then aD:= aD+6\*j2 else*

*if a='7' then aD:= aD+7\*j2 else*

*if a='8' then aD:= aD+7\*j2 else*

*if a='9' then aD:= aD+7\*j2;*

*j:=j+1 end;*

*Num\_2\_D:=aD*

*end;*

*function Num\_D\_16(aD:integer):string;*

*begin*

*a16:='';*

*while aD>0 do begin*

*aM:=aD mod 16;aD:=aD div 16;*

*if aM=0 then a16:='0'+a16 else*

*if aM=1 then a16:='1'+a16 else*

*if aM=2 then a16:='2'+a16 else*

*if aM=3 then a16:='3'+a16 else*

*if aM=4 then a16:='4'+a16 else*

*if aM=5 then a16:='5'+a16 else*

*if aM=6 then a16:='6'+a16 else*

*if aM=7 then a16:='7'+a16 else*

*if aM=8 then a16:='8'+a16 else*

*if aM=9 then a16:='9'+a16 else*

*if aM=10 then a16:='A'+a16 else*

*if aM=11 then a16:='B'+a16 else*

*if aM=12 then a16:='C'+a16 else*

*if aM=13 then a16:='D'+a16 else*

*if aM=14 then a16:='E'+a16 else*

*if aM=15 then a16:='F'+a16 end;*

*Num\_D\_16:=a16*

*end;*

*begin*

*Write('a2?');Read(a2);*

*aD:=Num\_2\_D(a2);*

*a16:=Num\_D\_16(aD);*

*WriteLn(a16)*

*end.*

Ця задача рівня ІІІ (обласного) етапу Всеукраїнської олімпіади зпрограмування.

*program Syst\_15\_D;uses Crt;*

*var a15,a:string;i,n:integer;aD,j,j15:real;*

*begin*

*Write('a15?');Read(a15);*

*n:=Length(a15);aD:=0;j:=0;*

*for i:=1 to n do begin*

*a:=Copy(a15,n-i+1,1);j15:=Power(15,j);*

*if a='0' then aD:= aD+0\*j15 else*

*if a='1' then aD:= aD+1\*j15 else*

*if a='2' then aD:= aD+2\*j15 else*

*if a='3' then aD:= aD+3\*j15 else*

*if a='4' then aD:= aD+4\*j15 else*

*if a='5' then aD:= aD+5\*j15 else*

*if a='6' then aD:= aD+6\*j15 else*

*if a='7' then aD:= aD+7\*j15 else*

*if a='8' then aD:= aD+8\*j15 else*

*if a='9' then aD:= aD+9\*j15 else*

*if a='A' then aD:=aD+10\*j15 else*

*if a='B' then aD:=aD+11\*j15 else*

*if a='C' then aD:=aD+12\*j15 else*

*if a='D' then aD:=aD+13\*j15 else*

*if a='E' then aD:=aD+14\*j15;*

*j:=j+1 end;*

*WriteLn(aD);*

*end.*

На закінчення наводиться програма, яка переводить числа із багатоцифрових систем числення у інші багатоцифрові системи числення із використанням ввод/виводу через текстові файли.

*Приклад*:

*Mrz.dat*: 12 16 73296664102529 (*dat*-файл);

*Mrz.res*: 73296664102529(12) = 4C36280289C71(16) (файл.*res*).

У файл *Mrz.dat* записані через пропуск основи систем числення введеного числа (12) та числа-результату (16) та саме число 73296664102529 (12), а у результуючий файл *Mrz.res* містить введене число та число-результат 73296664102529 (12) = 4C36280289C71 (16).

*Program Multi\_Systems­\_Long;*

*var i,u,r,j:integer;*

*s,v,w:string;*

*f:text;*

*function Add(st1,st2:string):string;{---------------------------- Add=st1+st2}*

*var i,l1,l2,pm:byte;code,s,s1,s2:integer;st:string;*

*begin*

*if st1[0]<st2[0] then begin st:=st1;st1:=st2;st2:=st end;*

*st1:='0'+st1;l1:=Length(st1);l2:=Length(st2);pm:=0;*

*for i:=l1-l2 downto 1 do st2:='0'+st2;*

*for i:=l1 downto 1 do begin*

*Val(st1[i],s1,code);Val(st2[i],s2,code);s:=s1+s2+pm;*

*st1[i]:=Chr((s mod 10)+48);pm:=s div 10 end;*

*if st1[1]='0' then Delete (st1,1,1);Add:=st1*

*end;{-----------------------------------------------------------------end Add}*

*function Sub(s1,s2:string):string;{----------------------------Sub=abs(s1-s2)}*

*var j,l1,l2,pm:byte;code,s\_1,s\_2,s:integer;*

*begin*

*l1:=Length(s1);l2:=Length(s2);pm:=0;*

*if s1=s2 then Sub:='0'*

*else begin*

*for j:=l1-l2 downto 1 do s2:='0'+s2;*

*for j:=l1 downto 1 do begin*

*Val(s1[j],s\_1,code);Val(s2[j],s\_2,code);*

*if s\_1>=s\_2+pm then begin s:=s\_1-(s\_2+pm);pm:=0 end*

*else begin s:=(s\_1+10)-(s\_2+pm);pm:=1 end;*

*s1[j]:=Chr(s+48) end;*

*while s1[1]='0' do Delete (s1,1,1);Sub:=s1 end*

*end;{-----------------------------------------------------------------end Sub}*

*function Mult1(st1,st2:string):string;{----------------------------Mult1}*

*var i,l1,l2,pm,s:byte;st:string;s1,s2,code:integer;*

*begin*

*st1:='0'+st1;l1:=Length(st1);l2:=Length(st2);pm:=0;*

*for i:=l1 downto 1 do begin*

*Val(st1[i],s1,code);Val(st2,s2,code);s:=s1\*s2+pm;*

*st1[i]:=Chr((s mod 10)+48);pm:=s div 10 end;*

*if st1[1]='0' then Delete(st1,1,1);Mult1:=st1*

*end;{---------------------------------------------------------------end Mult1}*

*function Mult(st1,st2:string):string;{---------------------------Mult=st1\*st2}*

*var i,j,l:byte;st0,st:string;*

*begin*

*st0:='';l:=Length(st2);*

*for i:=l downto 1 do begin*

*st:=Mult1(st1,Copy(st2,i,1));*

*for j:=1 to l-i do st:=st+'0';*

*st0:=Add(st,st0) end;*

*Mult:=st0;*

*end;{----------------------------------------------------------------end Mult}*

*function Comp(s1,s2:string):boolean;{-------------------Ї®аiў­п­­п st1 в  st2}*

*var j,l1,l2:byte;st:string;*

*begin l1:=Length(s1);l2:=Length(s2);*

*if l1<l2 then for j:=l2-l1 downto 1 do s1:='0'+s1*

*else for j:=l1-l2 downto 1 do s2:='0'+s2;*

*if s1>=s2 then Comp:=true else Comp:=false*

*end;{-----------------------------------------------end Comp}*

*procedure MoDiv(st1,st2:string;var stmod,stdiv:string);{--------------mod,div}*

*var i,l,k, j:integer;st\_1:string;bZero: boolean;*

*begin*

*st\_1:='';stdiv:='';i:=1;l:=Length(st1);*

*repeat*

*k:=0;if st2='' then break;*

*while not Comp(st\_1,st2) do begin*

*st\_1:=st\_1+st1[i];i:=i+1;stdiv:=stdiv+'0';*

*if i>Length(st2) then break;bZero := true;*

*if Length(st\_1)>Length(st2) then begin*

*For j := 1 to Length(st\_1)-1 do*

*if st\_1[j]<>'0' then bZero := false;*

*if (st\_1[Length(st\_1)]<>st2[Length(st2)]) and bZero then*

*begin Delete(st\_1,Length(st\_1),1);break end end end;*

*Delete (stdiv,Length(stdiv),1);*

*while st\_1[1]='0' do begin*

*If st\_1 = '' then break; Delete (st\_1,1,1) end;*

*while Comp(st\_1,st2) do begin*

*st\_1:=Sub(st\_1,st2);k:=k+1 end;*

*stmod:=st\_1;*

*if st\_1 = '' then stmod := '0';*

*stdiv:=stdiv+Chr(k+48);*

*while Length(stdiv)>Length(st1) do*

*Delete(stdiv,Length(stdiv),1);*

*until i>l;*

*if (Length(stdiv)<>1) and (stdiv[1]='0') then*

*while stdiv[1]='0' do Delete(stdiv,1,1);*

*end;{---------------------------------------------------------------end MoDiv}*

*function Stepin(st1:string;st2:integer):string;*

*var st:string;i:integer;*

*begin*

*if st2>2 then st:=Mult(st1,st1);*

*for i:=3 to st2 do*

*st:=Mult(st,st1);*

*if st2=0 then st:='1';*

*if st2=2 then st:=Mult(st1,st1);*

*if st2=1 then st:=st1;*

*Stepin:=st;*

*end;*

*procedure Inpt;*

*begin*

*Assign(f,'mrz1.dat');*

*Reset(f);*

*Read(f,u,r,s);*

*Delete(s,1,1);*

*Close(f);*

*end;*

*function toD(b,n:string):string;*

*var k,i:integer; a,a1:string;*

*begin*

*k:=Length(b);*

*for i:=1 to k do*

*begin*

*for j:=1 to 9 do*

*if b[i]=IntToStr(j) then a1:=Add(a1,Mult(IntToStr(j),Stepin(n,(k-i))));*

*if b[i]='A' then a1:=Add(a1,Mult('10',Stepin(n,k-i)));*

*if b[i]='B' then a1:=Add(a1,Mult('11',Stepin(n,k-i)));*

*if b[i]='C' then a1:=Add(a1,Mult('12',Stepin(n,k-i)));*

*if b[i]='D' then a1:=Add(a1,Mult('13',Stepin(n,k-i)));*

*if b[i]='E' then a1:=Add(a1,Mult('14',Stepin(n,k-i)));*

*if b[i]='F' then a1:=Add(a1,Mult('15',Stepin(n,k-i)));*

*end;*

*toD:=a1;*

*a1:='';*

*end;*

*function Dto(p,n:string):string;*

*var a,i,code:integer; b,c:string;*

*begin*

*while p<>'0' do*

*begin*

*MoDiv(p,n,c,p);*

*for i:=0 to 9 do*

*if c=IntToStr(i) then b:=c+b;*

*if c='10' then b:='A'+b;*

*if c='11' then b:='B'+b;*

*if c='12' then b:='C'+b;*

*if c='13' then b:='D'+b;*

*if c='14' then b:='E'+b;*

*if c='15' then b:='F'+b;*

*end;*

*Dto:=b;*

*b:='';*

*end;*

*procedure mrz;*

*begin*

*if u<>10 then w:=toD(s,IntToStr(u))*

*else w:=s;*

*if r<>10 then v:=Dto(w,IntToStr(r))*

*else v:=w;*

*end;*

*procedure Outp;*

*begin*

*Assign(f,'mrz1.res');*

*Rewrite(f);*

*Write(f,s,'(',u,') = ',v,'(',r,')');*

*Close(f);*

*end;*

*begin*

*Inpt;*

*Mrz;*

*Outp;*

*end.*

Ця програма рівня роботи МАН.