**Лабораторна робота №6. Масиви.**

1. Вимоги
   1. Розробник

* Бельчинська Катерина Юріївна
* студентка групи КІТ-320
* 02-nov-2020
  1. Індивідуальне завдання

Перетворити число (максимальне значення якого - 9999) в рядок. Наприклад,

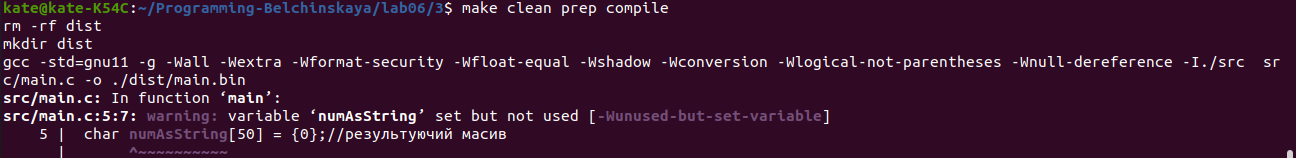
* 123 – “Сто двадцять три”
* 4321 – “Чотири тисячі триста двадцять один”.

1. Опис роботи

Код програми:

|  |
| --- |
| #define NUM 2334//задане число |
|  |  |
|  | int main() { |
|  | int num = NUM; |
|  | char numAsString[50] = {0};//результуючий масив |
|  | short size = 0; |
|  | //повідомляємо розряди чисел у строковому типі |
|  | char thousand[10] = {' ','t','h','o','u','s','a','n','d',' '}; |
|  | char hundred[9] = {' ','h','u','n','d','r','e','d',' '}; |
|  | char decade[8] = {' ','d','e','c','a','d','e',' '}; |
|  | //повідомляємо числа, які будемо використовувати як строки |
|  | char one[3] = {'o','n','e'}; |
|  | char two[3] = {'t','w','o'}; |
|  | char three[5] = {'t','h','r','e','e'}; |
|  | char four[4] = {'f','o','u','r'}; |
|  | char five[4] = {'f','i','v','e'}; |
|  | char six[3] = {'s','i','x'}; |
|  | char seven[5] = {'s','e','v','e','n'}; |
|  | char eight[5] = {'e','i','g','h','t'}; |
|  | char nine[4] = {'n','i','n','e'}; |
|  | int cond = 0; |
|  |  |
|  | for (int i = 10000; i > 0; i /= 10) {//довжина числа |
|  | if (num > 0) { |
|  | cond = num % i; |
|  | if (cond != num) { |
|  | size++; |
|  | } |
|  | num %= i; |
|  | } |
|  | } |
|  |  |
|  | num = NUM; |
|  | int iStart = 0; |
|  |  |
|  | if (size == 4) {//якщо число чотирьохзначне записуємо перше число строкою |
|  | num = NUM / 1000; |
|  | switch (num) { |
|  | case 1: |
|  | for (int t = 0; t < 3; t++) { |
|  | numAsString[iStart] = one[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 2: |
|  | for (int t = 0; t < 3; t++) { |
|  | numAsString[iStart] = two[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 3: |
|  | for (int t = 0; t < 5; t++) { |
|  | numAsString[iStart] = three[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 4: |
|  | for (int t = 0; t < 4; t++) { |
|  | numAsString[iStart] = four[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 5: |
|  | for (int t = 0; t < 4; t++) { |
|  | numAsString[iStart] = five[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 6: |
|  | for (int t = 0; t < 3; t++) { |
|  | numAsString[iStart] = six[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 7: |
|  | for (int t = 0; t < 5; t++) { |
|  | numAsString[iStart] = seven[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 8: |
|  | for (int t = 0; t < 5; t++) { |
|  | numAsString[iStart] = eight[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 9: |
|  | for (int t = 0; t < 4; t++) { |
|  | numAsString[iStart] = nine[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | } |
|  | for (int th = 0; th < 10; th++, iStart++) {//додаємо розряд даного числа |
|  | numAsString[iStart] = thousand[th]; |
|  | } |
|  |  |
|  | size--;//перебираємо числа на розряд нижчі |
|  | num = NUM % 1000; |
|  | } |
|  | if (size == 3) {// оприділяємо сотні числа |
|  | num = num / 100; |
|  | switch (num) { |
|  | case 0: |
|  | break; |
|  | case 1: |
|  | for (int t = 0; t < 3; t++) { |
|  | numAsString[iStart] = one[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 2: |
|  | for (int t = 0; t < 3; t++) { |
|  | numAsString[iStart] = two[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 3: |
|  | for (int t = 0; t < 5; t++) { |
|  | numAsString[iStart] = three[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 4: |
|  | for (int t = 0; t < 4; t++) { |
|  | numAsString[iStart] = four[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 5: |
|  | for (int t = 0; t < 4; t++) { |
|  | numAsString[iStart] = five[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 6: |
|  | for (int t = 0; t < 3; t++) { |
|  | numAsString[iStart] = six[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 7: |
|  | for (int t = 0; t < 5; t++) { |
|  | numAsString[iStart] = seven[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 8: |
|  | for (int t = 0; t < 5; t++) { |
|  | numAsString[iStart] = eight[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 9: |
|  | for (int t = 0; t < 4; t++) { |
|  | numAsString[iStart] = nine[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | } |
|  | for (int h = 0; h < 9; h++, iStart++) { |
|  | numAsString[iStart] = hundred[h]; |
|  | } |
|  | size--; |
|  | num = NUM % 100; |
|  | } |
|  |  |
|  | if (size == 2) {// записуємо десятки у строку |
|  | num = num / 10; |
|  | switch (num) { |
|  | case 0: |
|  | break; |
|  | case 1: |
|  | for (int t = 0; t < 3; t++) { |
|  | numAsString[iStart] = one[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 2: |
|  | for (int t = 0; t < 3; t++) { |
|  | numAsString[iStart] = two[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 3: |
|  | for (int t = 0; t < 5; t++) { |
|  | numAsString[iStart] = three[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 4: |
|  | for (int t = 0; t < 4; t++) { |
|  | numAsString[iStart] = four[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 5: |
|  | for (int t = 0; t < 4; t++) { |
|  | numAsString[iStart] = five[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 6: |
|  | for (int t = 0; t < 3; t++) { |
|  | numAsString[iStart] = six[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 7: |
|  | for (int t = 0; t < 5; t++) { |
|  | numAsString[iStart] = seven[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 8: |
|  | for (int t = 0; t < 5; t++) { |
|  | numAsString[iStart] = eight[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 9: |
|  | for (int t = 0; t < 4; t++) { |
|  | numAsString[iStart] = nine[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | } |
|  | for (int d = 0; d < 8; d++, iStart++) {//додаємо значення минулого числа - десятки |
|  | numAsString[iStart] = decade[d]; |
|  | } |
|  | size--; |
|  | } |
|  |  |
|  | if (size == 1) {//записуємо одиниці строками |
|  | num = NUM % 10; |
|  | switch (num) { |
|  | case 0: |
|  | break; |
|  | case 1: |
|  | for (int t = 0; t < 3; t++) { |
|  | numAsString[iStart] = one[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 2: |
|  | for (int t = 0; t < 3; t++) { |
|  | numAsString[iStart] = two[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 3: |
|  | for (int t = 0; t < 5; t++) { |
|  | numAsString[iStart] = three[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 4: |
|  | for (int t = 0; t < 4; t++) { |
|  | numAsString[iStart] = four[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 5: |
|  | for (int t = 0; t < 4; t++) { |
|  | numAsString[iStart] = five[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 6: |
|  | for (int t = 0; t < 3; t++) { |
|  | numAsString[iStart] = six[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 7: |
|  | for (int t = 0; t < 5; t++) { |
|  | numAsString[iStart] = seven[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 8: |
|  | for (int t = 0; t < 5; t++) { |
|  | numAsString[iStart] = eight[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | case 9: |
|  | for (int t = 0; t < 4; t++) { |
|  | numAsString[iStart] = nine[t]; |
|  | iStart++; |
|  | } |
|  | break; |
|  | } |
|  | numAsString[iStart] = '\0';//кінець результуючого масиву |
|  | } |
|  |  |
|  | return 0; |
|  | } |

Компіляція:



Відладка:



1. Варіанти використання

Програму доцільно використовувати для переводу числа у його наіменування

Висновок

Отже, програма переводить число, яке менше 9999, у слова

1. Загальне завдання

1

|  |
| --- |
| #define LEN 15 |
|  | int main() { |
|  | char zap = '\_'; |
|  | char strIn[] = "Ivanov"; |
|  |  |
|  | char strResult[LEN]; |
|  |  |
|  | int lenIn = 0;//кількість елементів заданого масиву |
|  |  |
|  | for (int i = 0; ; i++) {//дізнаємося величину масиву |
|  | if (strIn[i] == '\0') {//якщо елемент масиву '\0', то наступного елементу не існує |
|  | break; |
|  | } else if (strIn[i] != 0) { |
|  | lenIn++; |
|  | } |
|  | } |
|  | int lenZap = LEN - lenIn - 1;//кількість елементів в масиві, виділених на заповнювач |
|  | int halfLenZap = lenZap / 2;//кількість заповнювачей перед заданм рядком |
|  |  |
|  | for (int i = 0; i < halfLenZap; i++) {//в початок результуючого масиву вставимо необхідну кількість заповнювачей |
|  | strResult[i] = zap; |
|  | } |
|  |  |
|  | for (int i = halfLenZap; i < (halfLenZap + lenIn);) {//вставимо заданий на початку текст по центру результуючого масиву |
|  | for (int j = 0; j < lenIn; j++) { |
|  | strResult[i] = strIn[j]; |
|  | i++; |
|  | } |
|  | } |
|  |  |
|  | for (int i = (halfLenZap + lenIn); i < LEN - 1; i++) {//додаємо у результуючий масив заповнювач до його кінця |
|  | strResult[i] = zap; |
|  | } |
|  |  |
|  | strResult[LEN] = '\0';//масив повинен закінчуватися '\0' |
|  |  |
|  | return 0; |
|  | } |

2

|  |
| --- |
| #define PD 1//початок діапазону знаходження простих чисел |
|  | #define KD 200//кінець діапазону знаходження простих чисел |
|  | #define LEN 100//довжина результуючого масиву |
|  |  |
|  | int main() { |
|  | int prime[LEN] = {0};//результуючий масив з простих чисел спочатку повністю заповнений нулями |
|  | int j = 0; |
|  | for (int diap = PD; diap < KD; diap++ ) {//в заданому діапазоні вводимо змінну, щоб оприділяти чи є число простим |
|  | int flag = 0; |
|  | for (int i = 2; i < diap; i++) {//якщо число ділиться націло хоч на якесь число менше за нього, воно не може бути простим |
|  | if (diap % i == 0) { |
|  | flag = 0;//false |
|  | break; |
|  | } else { |
|  | flag = 1;//true |
|  | } |
|  | } |
|  | if (flag == 1) { |
|  | prime[j] = diap;//якщо число просте - вписуємо його у масив |
|  | j++; |
|  | } |
|  | } |
|  | return 0; |
|  | } |

4

|  |
| --- |
| int main() { |
|  | char strIn[] = "Hello World and others"; |
|  | int len = 0;//довжина масиву |
|  | int count = 0;//кількість слів |
|  |  |
|  | for (int i = 0; ; i++) {//знайдемо довжину масиву |
|  | if (strIn[i] == '\0') { |
|  | break; |
|  | } else { |
|  | len++; |
|  | } |
|  | } |
|  | for (int i = 0; i < len; i++) {//знайдемо кількість слів за допомогою останньої букви слова |
|  | if (strIn[i] != ' ' && (strIn[i + 1] == ' ' || strIn[i + 1] == '\0')) { |
|  | count++; |
|  | } |
|  | } |
|  | return 0; |
|  | } |

5

|  |
| --- |
| #define N 3 |
|  | int main() { |
|  | int arrayIn[N][N]; |
|  | for (int i = 0; i < N; i++) {//перебираємо елементи масиву |
|  | for (int j = 0;j < N; j++) { |
|  | arrayIn[i][j] = i \* N + j; |
|  | } |
|  | } |
|  | for (int i = 0; i < N; i++) {//за допомгою додаткової змінної k перебираємо строки заданої матриці |
|  | int k = arrayIn[i][0]; |
|  | for (int j = 0; j < N; j++) {//зміщуємо елементи масиву на 1 вправо |
|  | arrayIn[i][j] = arrayIn[i][j + 1]; |
|  | } |
|  | arrayIn[i][N - 1] = k;//результат прирівнюємо знов до масиву |
|  | } |
|  | return 0; |
|  | } |

6

|  |
| --- |
| #define N 3 |
|  | int main() { |
|  | int matrixIn[N][N];//дана матриця NxN |
|  | int matrixOut[N][N];//матриця, яка буде помножуватися |
|  |  |
|  | for (int i = 0; i < N; i++) {//обчислимо з даного масиву елементи матриці |
|  | for (int j = 0; j < N; j++) { |
|  | matrixIn[i][j] = i \* N + j; |
|  | } |
|  | } |
|  | for (int i = 0; i < N; i++) { |
|  | for (int j = 0; j < N; j++) { |
|  | matrixOut[i][j] = 0; |
|  | for (int k = 0; k < N; k++) {//вводимо додаткову змінну, щоб помножити по черзі кожну строку на кожний стовпчик |
|  | matrixOut[i][j] += matrixIn[i][k] \* matrixIn[k][j]; |
|  | } |
|  | } |
|  | } |
|  |  |
|  | return 0; |
|  | } |