МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ВЛАДИВОСТОКСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ВВГУ»)

ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И АНАЛИЗА ДАННЫХ
КАФЕДРА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И СИСТЕМ

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №6

по дисциплине
«Информатика и программирование»

Студент
гр. БИН-25-3 _____ Л.А. Сидоров

Ассистент
преподавателя _____ М.В. Водяницкий

Владивосток 2025

Задание

Выполнить задания и оформить отчет по стандартам ВВГУ.

Задание 1. Написать функцию, которая конвертирует время из одной величины в другую.

На вход подается:

- число (величина времени)
- исходная единица измерения
- единица измерения, в которую нужно перевести

Функция должна вернуть конвертированное значение

Задание 2. Пользователь делает вклад в банке в размере A рублей сроком на N лет

Процент по вкладу зависит от суммы и срока

Зависимость от суммы:

- каждые 10 000 рублей увеличивают ставку на 0.3%
- но суммарное увеличение не может превышать 5%
- минимальный вклад - 30 000 рублей

Зависимость от срока:

- первые 3 года - 3%
- от 4 до 6 лет - 5%
- более 6 лет - 2%

Необходимо написать функцию, которая рассчитывает прибыль пользователя без учета первоначально вложенной суммы

Используется сложный процент: каждый год процент начисляется на текущую сумму вклада

На вход подаются: сумма вклада и количество лет. Результат: сумма прибыли (не весь вклад, а только заработанные проценты)

Задание 3. Написать функцию для вывода всех простых чисел в заданном диапазоне. Нужно учитывать некорректные данные (например, начало больше конца или диапазон без простых чисел)

На вход подаются два числа: начало и конец диапазона (включительно). На выходе - список всех простых чисел или сообщение об ошибке

Задание 4. Реализовать функцию сложения двух матриц

При сложении двух матриц получается новая матрица того же размера, где каждый элемент - это сумма элементов с тем же индексом из двух исходных матриц

Ограничения:

- складывать можно только матрицы одинакового размера
- размер матрицы должен быть строго больше 2 (например, 3×3, 4×4 и т.д.)
- при нарушении условий нужно вывести сообщение об ошибке

На вход подаются:

- размер матрицы n (для квадратной матрицы n × n)
- элементы первой матрицы (по строкам, через пробел)
- элементы второй матрицы в таком же формате

Результат - новая матрица (в том же формате), либо сообщение об ошибке

Задание 5. Написать функцию, которая определяет, является ли строка палиндромом

Палиндром - это строка, которая читается одинаково слева направо и справа налево (обычно без учета пробелов, регистра и знаков препинания - эти правила нужно явно задать в своей реализации)

На вход подается строка. На выходе:

- Да, если это палиндром
- Нет, если это не палиндром

Содержание

1 Выполнение работы

1.1 Задание 1

Функция принимает на вход 3 параметра: time типа float, unit1 и unit2 типа char. Первым указывается время, вторым текущая единица измерения и третьм желаемая единица. Далее были заведены 3 константы с говорящими названиями для удобной конвертации. Следующим шагом обрабатываем входные данные на корректность, проверяем, что значение больше 0.

```c
#include <stdio.h>

float task1(float time, char unit1, char unit2)
{
  const int minutsInHour = 60;
  const int secondsInMinute = 60;
  const int secondsInHour = 3600;

  if (time <= 0) return 0;

  if (unit1 == 's')
  {
    if (unit2 == 'm') return (time / secondsInMinute);
    else if (unit2 == 'h') return (time / secondsInHour);
  }
  else if (unit1 == 'm')
  {
    if (unit2 == 's') return (time * secondsInMinute);
    else if (unit2 == 'h') return (time / secondsInHour);
  }
  else if (unit1 == 'h')
  {
    if (unit2 == 's') return (time * secondsInHour);
    else if (unit2 == 'm') return (time / minutsInHour);
  }
  return 0;
}
```

Рисунок 1 – Листинг программы для задания 1

Далее с помощью условий выясняем какая единица измерения дана и в какую из нее нужно перевести и переводим по соответствующим формулам. На рисунке 1 представлен код полученной программы.

1.2 Задание 2

Функция на вход принимает 2 значения: startSum типа float и years типа int. Первым указывается вносимая сумма, а вторым - срок, на который хотим создать вклад. После проверяем данные на корректность: сумма должна быть больше 0 и кол-во лет тоже больше 0, в противном случае возвращаем 0. Далее заводим такой список переменных:

- sum типа float, которая будет хранить накопленную сумму.
- rate типа float, которая будет хранить коэффициент на текущий период.

- bonusRate типа float, которая будет хранить бонусный коэффициент за каждые 10 тысяч вклада.

- firstRate типа float, которая будет хранить коэффициент для первых 3-ех лет вклада.

- seconRate типа float, которая будет хранить коэффициент для следующих 3-ех лет вклада.

- thirdRate типа float, которая будет хранить коэффициент для всех последующих лет вклада.

- maxBonusRate типа float, которая будет хранить максимальное значение бонусного коэффициента.

```
#include <stdio.h>
#include <math.h>

float calculateFinalSum(float sum, float rate, int years)
{
  return (sum * pow(1 + rate, years));
}

float task2(float startSum, int years)
{
  if (startSum <= 0) return 0;
  else if (years <= 0) return 0;

  float sum = startSum;
  float rate = 0;
  float bonusRate = 0;
  float firstRate = 0.03f; // первые 3 года
  float secondRate = 0.05f; // последующие 2 года
  float thirdRate = 0.02f; // начиная с ого6- года
  float maxBonusRate = 0.05f; //Максимальный бонусный
    коэффициент

  bonusRate = (int)startSum / 10000 * 0.003f;
  if (bonusRate > maxBonusRate) bonusRate = maxBonusRate;

  if (years - 3 <= 0) return calculateFinalSum(sum,
    firstRate + bonusRate, years) - startSum;
  else
  {
    sum = calculateFinalSum(sum, firstRate + bonusRate, 3);
    years -= 3;
  }
  if (years - 3 <= 0)
  {
    return calculateFinalSum(sum, secondRate + bonusRate,
   years) - startSum;
  }
  else
  {
    sum = calculateFinalSum(sum, secondRate + bonusRate, 3);
    years -= 3;
  }
  return calculateFinalSum(sum ,thirdRate + bonusRate, years
   ) - startSum;
}
```

Рисунок 2 – Листинг программы для задания 2

Следующим шагом считаем бонусный коэффициент и в случае, если он превышет макимальное значение, присваиваем максимальное значение. После по формуле: sum * (1 + rate) ŷears считаем суммы по соответствующим периодам. В конце от итоговой суммы отнимаем начальную и выводим значение. На рисунке 2 представлен код полученной программы.

1.3 Задание 3

Процедура на вход принимает верхнюю и нижнюю границы диапазона, простые числа которого необходимо вывести. Параметры принимаются в переменные low типа int и high типа int соответственно. Первым делом заводим счетчик cntSimpleNums типа int для подсчета найденных простых чисел. Далее проверяем данные на корректность: нижняя граница не может быть больше верхней, а верхняя не может быть отрицательной.

```c
#include <stdio.h>

int isSimple(int num)
{
  if (num <= 0) return 0;

  int cntOfDivides = 0;
  int highestDivider = sqrt(num);
  for (int j = 1; j <= highestDivider; j += 2)
  {
    if (num % j == 0) ++cntOfDivides;
    if (cntOfDivides > 1) return 0;
  }
  return 1;
}

void task3(int low, int high)
{
  int cntSimpleNums = 0;

  if (low > high)
  {
    printf("Error! lowValue > highValue! \n");
    return;
  }
  else if (low <= 0)
  {
    printf("Error! lowValue <= 0! \n");
    return;
  }

  if (low <= 3)
  {
    for (int i = 1; i <= 3; i++)
    {
      printf("%d ", i);
      cntSimpleNums++;
    }
    low = 5;
  }

  if ((low & 1) == 0) low += 1;

  for (int i = low; i <= high; i += 2)
  {
    if (isSimple(i))
    {
      printf("%d ", i);
      cntSimpleNums++;
    }
  }
    if (cntSimpleNums == 0) printf("No simple nums! \n");
    printf("\n");
}
```

Рисунок 3 – Листинг программы для задания 3

После проверяем, если нижняя граница < 3, то выведем заведомо простые числа, дабы не усложнять алгоритм, поскольку дальше, к примеру та же 2 вообще не будет существовать для программы, поскольку она четная, а за исключением ее, все остальные четные числа не простые. В конце проверяем осташиеся числа на простоту и выводим

найденные или сообщение о том, что в заданном диапазоне нет таких чисел. На рисунке 3 представлен код программы.

1.4 Задание 4

Заводим 2 переменные типа int - numOfStr, что будет хранить кол-во строк, и numOfCol, что будет хранить кол-во столбцов. Далее просим пользователя ввести соответствующие параметры. После ввода проверяем их на корректность: оба должны быть больше 0. После чего объявляем 3 двумерных массива одинакового размера, соответствующего параметрам. matrix0 хранит первую матрицу, а matrix1 - вторую, а resMatrix служит для результата сложения матриц.

```c
#include <stdio.h>

void printMatrix(int* matrix, int numOfStr, int numOfCol)
{
  for (int i = 0; i < numOfStr; i++)
  {
    for (int j = 0; j < numOfCol; j++)
    {
      printf("%d ",matrix[i, j]);
    }
    printf("\n");
  }
}

void inputMatrixValues(int* matrix,int numOfStr, int
   numOfCol)
{
  for (int i = 0; i < numOfStr; i++)
  {
    printf("Введите элементы строки %d через пробел. Затем
    нажмите ENTER. \n", i+1);

    for(int j = 0; j < numOfCol; j++)
    {
      scanf("%d", &matrix[i, j]);
    }
  }
}

void task4()
{
  int numOfStr = 0;
  int numOfCol = 0;

  printf("Введите колво- строк в матрицае: ");
  scanf("%d", &numOfStr);

  printf("Введите колво- столбцов в матрицае: ");
  scanf("%d", &numOfCol);

  if (numOfCol <= 0 || numOfStr <= 0) printf("Error!
    numOfCol <= 0 or numOfStr <= 0!");

  int   matrix0[numOfStr][numOfCol];
  int   matrix1[numOfStr][numOfCol];
  int resMatrix[numOfStr][numOfCol];

  inputMatrixValues(*matrix0 ,numOfStr, numOfCol);
  inputMatrixValues(*matrix1 ,numOfStr, numOfCol);

  for (int i = 0; i < numOfStr; i++)
  {
    for (int j = 0; j < numOfCol; j++)
    {
      resMatrix[i][j] = matrix0[i][j] + matrix1[i][j];
    }
  }

  printMatrix(*resMatrix, numOfStr, numOfCol);
}
```

Рисунок 4 – Листинг программы для задания 4

Затем просим пользователя ввести значения соответствующих матриц. А в конце с помощью вложенных циклов for складываем соответствующие элементы матрицы и запи-

сываем в соответствующую ячейку результирующей матрицы. Последник шагом выводим результат. На рисунке 4 представлен код решения.

1.5 Задание 5

Функция на вход принимает строку и ее длинну. Далее заводим массив для новой строки, которая появится после удаления пробелов и переменную, в которой будет хранится индекс последнего символа новой строки. После удаляем все пробелы и запоминаем индекс последнего символа.

```
#include <stdio.h>

void printStr(char* str, int length)
{
  for (int i = 0; i < length; i++) printf("%c", str[i]);
  printf("\n");
}

int findEndStr(char* str, int length)
{
  for (int i = 0; i < length; i++)
  {
    if (str[i] == '\0') return i-1;
  }
  return length-1;
}

void deleteAllSpacesInStr(char* str, int length, char*
   emptyStrRes)
{
  int cnt = 0;
  for(int i = 0; i < length; i++)
  {
    if (str[i] == ' ') continue;
    else if (str[i] == '\0') return;
    emptyStrRes[cnt++] = str[i];
  }
}

void task5(char* startStr, int length)
{
  char str[length];
  int indexOfEndStr;

  deleteAllSpacesInStr(startStr, length, str);
  indexOfEndStr = findEndStr(str, length);

  for (int i = 0; i <= indexOfEndStr/2; i++)
  {
    printf("%c %c \n", str[i], str[indexOfEndStr-i]);
    if (str[i] != str[indexOfEndStr-i])
    {
      printf("Слово - не палиндром. !p \n");
      return;
    }
  }
  printf("Слово - палиндром. p \n");
}
```

Рисунок 5 – Листинг программы для задания 5

Затем с помощью цикла for запускаем сравнение i-ого и indexOfEndStr-i символов строки. В сулчае неравенства выводим сообщение, что слово не палиндром, и завершаем программу. На рисунке 5 представлен код программы.