Отчет по лабораторной работе № 1 «Введение в С. Microsoft Visual Studio»			
дата	Оценка (max 3)	Бонус за сложность	подпись

Цели работы:

-Ознакомление с базовыми возможностями и средствами языка С

Задачи работы:

-знакомство с основами С

-разработка примеров простейших программ на С

Краткий конспект теоретической части (ответы на контрольные вопросы)

Функция main() - это начальная точка выполнения для всех программ на языках С и
С++
Функция prinf позволяет выводить информацию на экран при программировании в
консольном режиме
Цикл for - цикл с постусловием — цикл, в котором условие проверяется после
выполнения тела цикла.

Задание 1

Запустите программу “Hello, World!”.

Исходный код

```
#include <stdio.h>

int main() {
    printf("Hello, World!\n");
    return 0;
}
```

Результат выполнения

```
gcc 1_1.c
./a.out
Hello, World!
```

Задание 2

Выведите следующую картинку с помощи функции printf:

****

* <Ваше имя> *

****

Исходный код

```
#include <stdio.h>

int main() {
    printf("**********\n");
    printf("***Никита Куликов***\n");
    printf("**********\n");
}
```

Результат выполнения

```
**********
***Никита Куликов**
**********
```

Задание 3

Напишите программу, печатающую таблицу умножения для чисел от 0 до 9 в десятичной системе счисления.

Исходный код

```
#include <stdio.h>

int main() {
    for (int i = 1; i < 10; i++) {
        for (int j = 1; j < 10; j++) {
            printf("%3d", i * j);
        }
        printf("\n");
    }
}
```

Результат выполнения

```
  1  2  3  4  5  6  7  8  9
  2  4  6  8 10 12 14 16 18
  3  6  9 12 15 18 21 24 27
  4  8 12 16 20 24 28 32 36
  5 10 15 20 25 30 35 40 45
  6 12 18 24 30 36 42 48 54
  7 14 21 28 35 42 49 56 63
  8 16 24 32 40 48 56 64 72
  9 18 27 36 45 54 63 72 81
```

Контрольные вопросы

1. Функция main?
2. Функция printf?
3. Цикл for?

Отчет по лабораторной работе № 2 «Основы С»			
дата	Оценка (max 3)	Бонус за сложность	подпись

Цели работы:

-Ознакомление с базовыми возможностями и средствами языка С

Задачи работы:

-знакомство с основами С

-разработка примеров простейших программ на С

Краткий конспект теоретической части (ответы на контрольные вопросы)

Процесс создания программного обеспечения: структура, согласно которой построена
разработка программного обеспечения (ПО)
Компиляция - трансляция программы, составленной на исходном языке высокого уровня,
в эквивалентную программу на низкоуровневом языке, близком машинному коду
(абсолютный код, объектный модуль, иногда на язык ассемблера), выполняемая
компилятором.
Переменные - в языках программирования именованная часть памяти, в которую
могут помещаться разные значения переменной
Функции - отдельная система (подсистема, подпрограмма), на вход которой
поступают управляющие воздействия в виде значений аргументов. На выходе системы
получаем результат выполнения программы, который может быть как скалярной
величиной, так и векторным значением.
Константы - некоторая величина, не изменяющая своё значение в рамках
рассматриваемого процесса.
Определение и объявление: При объявлении переменной, функции или даже класса все,
что вы делаете, это говорите компилятору, что есть что-то с определенным именем и
определенного типа. Компилятор может обрабатывать большинство (но не все)
использований этого имени, без необходимости полного определения этого имени.
Объявление значения, не определяя его, позволяет писать код, понятный компилятору,
опуская дополнительные детали

Задание 1

Напишите программу, выводящую каждое новое слово с новой строки.

Исходный код

```
#include <stdio.h>

int main() {
    char tmpChar = NULL;
    while (tmpChar != '\n') {
        tmpChar = getchar();
        if (tmpChar != ' ')
            putchar(tmpChar);
        else
            putchar('\n');
    }
}
```

Результат выполнения

```
Привет, Иноземец! Эта строка будет разбита по словам!
Привет,
Иноземец!
Эта
строка
будет
разбита
по
словам!
```

Задание 2

Напишите программу, которая заменяет символы табуляции, перехода на новую строку и остальные управляющие последовательности на \t, \n и т.д.

Исходный код

```
#include <stdio.h>

int main() {
    char tmpChar = NULL;
    while (tmpChar != '*') {
        tmpChar = getchar();
        switch (tmpChar){
            case '\n':
                putchar('/');
                putchar('n');
                break;
            case '\t':
                putchar('/');
                putchar('t');
                break;
            default:
                putchar(tmpChar);
        }
    }
}
```

Результат выполнения

Итак, это строка
Итак, это строка/n

Задание 3

Напишите программу, для подсчета количества пробелов, табуляций и символов перехода на новую строку.

Исходный код

```
#include <stdio.h>

int main() {
    int counterForSpace = 0, counterForTabs = 0, counterForLine = 0;
    char tmpChar = NULL;
    while (tmpChar != '*') {
        tmpChar = getchar();
        switch (tmpChar) {
            case '\n':
                counterForLine++;
                break;
            case '\t':
                counterForTabs++;
                break;
            case ' ':
                counterForSpace++;
                break;
        }
    }
    printf("Количество пробелов: %d\n", counterForSpace);
    printf("Количество табов: %d\n", counterForTabs);
    printf("Количество линий: %d\n", counterForLine);
    printf("Количество 'особых' символов: %d\n", counterForLine + counterForSpace
+ counterForTabs);
}
```

Результат выполнения

```
Привет, это форматированный текст!
Возможно, это кого-то удивит, но тут есть пробелы
Но табов нет :((((
*
Количество пробелов: 13
Количество табов: 0
Количество линий: 3
Количество 'особых' символов: 16
```

Контрольные вопросы

4. Процесс создания программного обеспечения?
5. Компиляция?
6. Переменные?
7. Функции?
8. Константы?
9. Объявления и определения?

Отчет по лабораторной работе № 3 «Типы данных, операции и выражения»			
дата	Оценка (max 3)	Бонус за сложность	подпись

Цели работы:

-Ознакомление с типами данных, операциями и выражениями С

Задачи работы:

-знакомство с основными типами данных, операциями и выражениями С

-разработка примеров простейших программ на С

Краткий конспект теоретической части (ответы на контрольные вопросы)

Переменные - Перемéнная в императивном программировании — поименованная, либо
адресуемая иным способом область памяти, адрес которой можно использовать
для осуществления доступа к данным.
Типы данных и их размеры- char (1 байт), int (2 байта), long (4 байта), long long (8 байт),
float(в стандарте не описан, обычно 64 бит), double (обычно 64 бит), long double (80 бит)
Символьная константа - Символьная константа – это некоторый символ алфавита,
заключенный в одиночные кавычки
Константное выражение - Константное выражение -- это выражение, состоящее из
одних констант. Такие выражения обрабатываются во время компиляции, а не при
прогоне программы, и соответственно могут быть использованы в любом месте,
где можно использовать константу, как, например в
Строковая константа - это нуль или более символов Unicode, заключенных в
одинарные или двойные кавычки.
Перечислимый тип - в программировании тип данных, чьё множество значений
представляет собой ограниченный список идентификаторов.
Виды операций и их приоритет – так же как и в арифметике

Задание 1

Напишите программу, которая преобразует строку шестнадцатеричных цифр в ее целочисленный эквивалент. Допустимые символы: 0-9, a-f, A-F.

Исходный код

```
int charToInt(char numeric) {
    if (numeric >= '0' && numeric <= '9')
        return numeric - '0';
    else if (numeric >= 'A' && numeric <= 'F')
        return numeric - 'A' + 10;
    else if (numeric >= 'a' && numeric <= 'f')
        return numeric - 'a' + 10;
    else return 0;
}

long long getLongFromHEX(size_t size, char *arr) {
    long long returnValue = 0;
    for (int i = (int) (size - 1), j = 0; i > -1; i--, j++) {
        returnValue += charToInt(arr[i]) *  pow(16.0, (double) j);
    }
    return returnValue;
}
```

Результат выполнения

```
B3A73CF8186
1234567898765
```

Задание 2

Напишите программу, удаляющую из строки s1 все символы, которые содержит строка s2. Удаление символов оформить в виде функции.

Исходный код

```
void removeSymbolsFromString(size_t s1Size, size_t s2Size, size_t outputSize,
char *s1, char *s2, char *outputArray) {
    outputSize = 0;
    bool contains = false;

    for (int i = 0; i < s1Size; i++) {
        for (int j = 0; j < s2Size && !contains; j++)
            if (s2[j] == s1[i])
                contains = true;
        if (!contains)
            outputArray[outputSize++] = s1[i];
        contains = false;
    }
    outputArray[outputSize] = '\0';
}
```

Результат выполнения

```
Privet!
asdfvg
Priet!
```

Задание 3

Напишите программу, которая бы разворачивала сокращенную запись наподобие a-z в строке s1 в полный список abc...xyz в строке s2. Учитывайте буквы в любом регистре, цифры.

Исходный код

```
int calculateNewSize(size_t stringSize, char *string) {
    int newSize = 0;
    for (int i = 0; i < stringSize; i++) {
        newSize++;
        if (string[i] == '-')
            newSize += string[i + 1] - string[i - 1] - 2;
    }
    return newSize + 1;
}

char* getFullString(size_t stringSize, size_t string2Size, char *string1, char *string2) {
    string2Size = (size_t) calculateNewSize(stringSize, string1);
    string2 = malloc(sizeof(char) * string2Size);
    int curPos = 0;

    for (int i = 0; i < stringSize; i++) {
        if (string1[i] != '-')
            string2[curPos++] = string1[i];
        else
            for (char ch = (char) (string1[i - 1] + 1); ch < string1[i + 1]; ch++)
                string2[curPos++] = ch;
    }
    string2[curPos] = '\0';
    return string2;
}
```

Результат выполнения

```
a-zA-Z0-9
abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789
```

Контрольные вопросы

1. Переменные?
2. Типы данных и их размеры?
3. Символьная константа?
4. Константное выражение?
5. Строковая константа?
6. Перечислимый тип?
7. Виды операций и их приоритет?

Отчет по лабораторной работе № 4 «Управляющие конструкции и простейшие алгоритмы»			
дата	Оценка (max 3)	Бонус за сложность	подпись

Цели работы:

-Ознакомление с типами данных, операциями и выражениями С

Задачи работы:

-знакомство с основными типами данных, операциями и выражениями С

-разработка примеров простейших программ на С

Краткий конспект теоретической части (ответы на контрольные вопросы)

Условные конструкции - Условные конструкции - один из базовых компонентов
многих языков программирования, которые направляют работу программы по одному
из путей в зависимости от определенных условий.
Циклы - разновидность управляющей конструкции в высокоуровневых языках
программирования, предназначенная для организации многократного исполнения
набора инструкций.
Двоичный поиск - классический алгоритм поиска элемента в отсортированном
массиве (векторе), использующий дробление массива на половины. Используется в
информатике, вычислительной математике и математическом программировании.
Функции - отдельная система (подсистема, подпрограмма), на вход которой поступают
управляющие воздействия в виде значений аргументов. На выходе системы получаем
результат выполнения программы, который может быть как скалярной величиной, так и
векторным значением.
Рекурсия - определение, описание, изображение какого-либо объекта или процесса
внутри самого этого объекта или процесса, то есть ситуация, когда объект является
частью самого себя. Термин «рекурсия» используется в различных специальных
областях знаний — от лингвистики до логики, но наиболее широкое применение
находит в математике и информатике.

Задание 1

Напишите программу, реализующую алгоритм сортировки пузырьком.

Исходный код

```
#include <stdbool.h>
#include <glob.h>
#include "stdio.h"
void bubbleSorting(size_t arraySize, int *array) ;

int main(){
    int array[8] = {0,1,2,3,4,5,6,7};

    for(int i = 0; i< 8; i++)
        printf("%d ", array[i]);
    printf("\n");
    bubbleSorting(8,array);

    for(int i = 0; i< 8; i++)
        printf("%d ", array[i]);
}

void bubbleSorting(size_t arraySize, int *array) {
    bool isSorting = true; // Метка о том, что в последнем прогоне было всплывание
    int tmp = 0;

    while (isSorting) {
        isSorting = false;
        for (int i = 1; i < arraySize; i++)
            if (array[i] > array[i - 1]) {
                tmp = array[i - 1];
                array[i - 1] = array[i];
                array[i] = tmp;
                isSorting = true;
            }
    }
}
```

Результат выполнения

```
0 1 2 3 4 5 6 7
7 6 5 4 3 2 1 0
```

Задание 2

Напишите программу, реализующую алгоритм сортировки Шелла.

Исходный код

```
#include <stdbool.h>
#include <glob.h>
#include "stdio.h"

void shellSorting(int arraySize, int *array) {
    int deltaArray[9] = {0, 1, 4, 10, 23, 57, 132, 301, 701};
    int deltaPos = 8;
    for (deltaPos = 8; deltaPos > 0 && deltaArray[deltaPos] > arraySize - 1; deltaPos--);

    int currentDelta = 0;
    for (currentDelta = deltaArray[deltaPos--]; currentDelta > 0; currentDelta =
deltaArray[deltaPos--]) {
        bool isSorting = true;
        int tmp = 0;

        while (isSorting) {
            isSorting = false;
            for (int i = currentDelta; i < arraySize; i++)
                if (array[i] > array[i - currentDelta]) {
                    tmp = array[i - currentDelta];
                    array[i - currentDelta] = array[i];
                    array[i] = tmp;
                    isSorting = true;
                }
        }
    }
}
```

Результат выполнения

```
0 1 2 3 4 5 6 7
7 6 5 4 3 2 1 0
```

Задание 3

Напишите программу, реализующую алгоритм быстрой сортировки.

Исходный код

```
void qsort(int *array, int start, int end) {
    int i = start;
    int j = end;
    int middle = array[(start + end) / 2];
    int tmp;

    do {
        while (array[i] > middle) i++;
        while (array[j] < middle) j--;

        if (i <= j) {
            if (array[i] < array[j]) {
                tmp = array[i];
                array[i] = array[j];
                array[j] = tmp;
            }
            i++;
            j--;
        }
    } while (i <= j);

    if (i < end)
        qsort(array, i, end);
    if (start < j)
        qsort(array, start, j);
}
```

Результат выполнения

```
0 1 2 3 4 5 6 7
7 6 5 4 3 2 1 0
```

Задание 4

В программе задан отсортированный массив целых чисел A. Пользователь в консоли вводит целые числа, каждое число с новой строки. Задача: написать программу, которая будет проверять есть ли число, которое ввел пользователь, в массиве A. При написании программы воспользоваться алгоритмом двоичного (бинарного) поиска.

Исходный код

```
#include <glob.h>
#include <stdlib.h>
#include "stdio.h"

int binarySearch(int n, size_t size,
 int *sortedArray);

int main() {
   int array[8] = {0, 1, 2, 3, 4, 5, 6, 7};
   int number = 0;
   while (number != -1) {
      scanf("%d", &number);

      int result = binarySearch(number,
8, array);

      if (array[result] == number)
         printf("OK!\n");
      else printf("NO!\n");
   }
   return 0;
}
```

```
int binarySearch(int n, size_t size, int
*sortedArray) {
   int start = 0, end = 0, curPos = 1;

   // Бежим степенью двойки до
ближайшего значения, больше искомого
   while (curPos < size &&
sortedArray[curPos] < n)
      curPos *= 2;;
   if (curPos > size)
      curPos = (int) size - 1;

   if (curPos == 1)
      start = 0;
   else start = curPos / 2;
   end = curPos;

   // Стандартный бинарный поиск
   do {
      curPos = (end - start) / 2;
      if (n > sortedArray[start + curPos])
         start = start + curPos;
      else end = start + curPos;
   } while (end - start > 1);

   if (end == size)
      return end - 1;
   else if (abs(sortedArray[start] - n) >
abs(sortedArray[end] - n))
      return end;
   else return start;
}
```

```
1
OK!
2
OK!
10
NO!
-1
NO!
```

Контрольные вопросы

1. Недостатки сортировки пузырьком?
2. Преимущества и недостатки сортировки Шелла?
3. Преимущества и недостатки быстрой сортировки?
4. Особенности алгоритма двоичного (бинарного) поиска?