Класичний приватний університет

Кафедра програмування та інформаційних технологій

Борю С.Ю

МЕТОДИЧНІ ВКАЗІВКИ
ДО ВИКОНАННЯ КОНТРОЛЬОНОЇ РОБОТИ З ДИСЦИПЛІНИ

Об`єктно-орієнтоване програмування
Перша частина

для студентів денного і заочного відділення спеціальності
"Програмне забезпечення автоматизованих систем" та
„Системний аналіз та управлення”

Запоріжжя, 2007

Методические указания по выполнению контрольной работы по курсу «об`єктно - орієнтоване програмування» для студентів денного і заочного відділення спеціальності "Програмне забезпечення автоматизованих систем" та „Системний аналіз та управлення” /сост. Борю С.Ю. - Запорожье: Класичний приватний університет, 2007 г. – с.

Данные методические указания включают примеры расчетов, варианты контрольной работы и рекомендуемые источники, которые можно использовать при выполнении работы.

Целью контрольной работы является закрепление и углубление знаний, полученных на лекциях, практических занятиях и в процессе самостоятельного изучения материала по курсу «об`єктно - орієнтоване програмування».

Контрольная работа предназначена для проверки теоретических и практических знаний студентов; для отработки навыков самостоятельного изучения предмета, работы с литературой и программным обеспечением, умение логично и последовательно излагать усвоенный материал.

Содержание

Общие требования к выполнению контрольной работы.

Для выполнения контрольной работы необходимо:

1) изучить теоретический материал;
2) ознакомиться с приведенным примером;
3) выбрать вариант по следующей формуле – номер варианта = остаток от деления номера Вашей заченой книжке на число 24.
4) выполнить работу;
5) оформить отчет по работе, используя текстовый процессор MS Word;
6) защита контрольной работы предполагает наличие электронного варианта.

Требования к оформлению отчета по контрольной работе.

1. Содержание работы оформить как автоматическое оглавление с указанием страниц и заполнителя.
2. Предусмотреть обычные сноски в тексте на использованные источники (литература, программное обеспечение).
3. Список источников оформить как нумерованный список.
4. В работе предусмотреть следующие разделы:
   - Титульный лист,
   - Содержание,
   - Теоретические сведения,
   - Порядок выполнения работы,
   - Список использованных источников.
5. Создать следующие колонтитулы для разделов:
   - верхний - слева Ваша ФИО, справа - название раздела;
   - нижний – слева номер варианта, справа - нумерация страниц;
   - титульный лист – колонтитулы отсутствуют.
6. Теоретические сведения оформляются в текстовом процессоре Word, с описанием типов и форматов данных, видов ссылок, формул, синтаксиса функций, построения диаграмм.
7. Описать порядок выполнения работы.

Задание 1. "Использование основных операторов языка Си"

Цель : Получение навыков в выборе и использовании операторов Си++; знакомство с итерационными процессами.

1.2 Краткие теоретические сведения

Операторы управления работой программы называют управляющими конструкциями программы. К ним относят:

- составные операторы;
- операторы выбора;
- операторы циклов;
- операторы перехода.

Составные операторы

К составным операторам относят собственно составные операторы и блоки. В обоих случаях это последовательность операторов, заключенная в фигурные скобки. Блок отличается от составного оператора наличием определений в теле блока. Например:

```
{
n++;                    это составной оператор
summa+=n;
}

{
int n=0;
n++;                    это блок
summa+=n;
}
```

Операторы выбора

Операторы выбора - это условный оператор и переключатель. Условный оператор имеет полную и сокращенную форму.

```
if ( <выражение-условие> ) <оператор>;          //сокращенная форма
```

В качестве <выражения-условия> могут использоваться арифметическое выражение, отношение и логическое выражение. Если значение <выражения-условия> отлично от нуля (т. е. истинно), то выполняется оператор. Например:

```
if (x<y&&x<z)min=x;
```

```
if ( <выражение-условие> ) <оператор1>;          //полная форма
else  <оператор2>;
```

Если значение <выражения-условия> отлично от нуля, то выполняется оператор1, при нулевом значении <выражения-условия> выполняется оператор2.Например:

```
if (d>=0)
{
x1=(-b-sqrt(d))/(2*a);
x2=(-b+sqrt(d))/(2*a);
cout<< “\nx1=”<<x1<<“x2=”<<x2;
}
else cout<<“\nРешения нет”;
```

Переключатель определяет множественный выбор.

```
switch (<выражение>)
{
case <константа1> : <оператор1 >;
case <константа2> : <оператор2 >;
. . . . . .
default: <операторы>;
```

При выполнении оператора switch, вычисляется выражение, записанное после switch и его значение последовательно сравнивается с константами, которые записаны следом за case. При первом же совпадении выполняются операторы помеченные данной меткой. Если выполненные операторы не содержат оператора перехода, то далее выполняются операторы всех следующих вариантов, пока не появится оператор перехода или не закончится переключатель. Если значение выражения, записанного после switch не совпало ни с одной константой, то выполняются операторы, которые следуют за меткой default. Метка default может отсутствовать.

Пример:

```
switch ( number )
{
case 1 :  cout<< “число=1”;break;
case 2 :  cout<< “2 * 2”<<number * number;
case 3 :  cout<< “3 * 3”<<number * number; break;
case 4 :  cout<< number<<“- это замечательное число”; break;
default: cout<< “Конец работы программы”;
}
```

Операторы циклов

1. Цикл с предусловием:

```
while (<выражение-условие>)
<тело_цикла> ;
```

В качестве <выражения-условия> чаще всего используется отношение или логическое выражение. Если оно истинно, т. е. не равно 0, то тело

цикла выполняется до тех пор пока <выражение-условие> не станет ложным.

2. Цикл с постусловием:

do
<тело_цикла>;
while (<выражение-условие>);

Тело цикла выполняется до тех пор, пока <выражение-условие> истинно.

3. Цикл с параметром:

for (<выражение_1>;<выражение-условие>;<выражение_3>)
тело_цикла;

<Выражение_1> и <выражение_3> могут состоять из нескольких выражений, разделенных запятыми. <Выражение_1> - задает начальные условия для цикла (инициализация).<Выражение-условие> определяет условие выполнения цикла, если оно не равно 0, цикл выполняется, а затем вычисляется значение <выражения_3>. <Выражение_3> - задает изменение параметра цикла или других переменных (коррекция). Цикл продолжается до тех пор, пока <выражение-условие> не станет равно 0. Любое выражение может отсутствовать, но разделяющие их « ; » должны быть обязательно.

Примеры использования цикла с параметром.

1) Уменьшение параметра:

for (n=10; n>0; n--)
{ <тело цикла>};

2) Изменение шага корректировки:

for (n=2; n>60; n+=13)
{ <тело цикла>};

3) Возможность проверять условие отличное от условия, которое налагается на число итераций:

for (num=1;num*num*num<216; num++)
{ <тело цикла>};

4) Коррекция может осуществляться не только с помощью сложения или вычитания:

for (d=100.0; d<150.0;d*=1.1)
{ <тело цикла>};
for (x=1;y<=75;y=5*(x++)+10)
{ <тело цикла>};

5) Можно использовать несколько инициализирующих или корректирующих выражений:

for (x=1, y=0; x<10;x++;y+=x);

Операторы перехода

Операторы перехода выполняют безусловную передачу управления.

1) break - оператор прерывания цикла.

```
{
< операторы>
if (<выражение_условие>) break;
<операторы>
}
```

Т. е. оператор break целесообразно использовать, когда условие продолжения итераций надо проверять в середине цикла.

Пример:

```
// ищет сумму чисел вводимых с клавиатуры до тех пор, пока не будет введено 100 чисел или 0
for(s=0, i=1; i<100;i++)
{
cin>>x;
if( x==0) break; // если ввели 0, то суммирование заканчивается
s+=x;
}
```

2) continue - переход к следующей итерации цикла. Он используется, когда тело цикла содержит ветвления.

Пример:

```
//ищет количество и сумму положительных чисел
for( k=0,s=0,x=1;x!=0;)
{
cin>>x;
if (x<=0) continue;
k++;s+=x;
}
```

1.2 Постановка задачи

Используя оператор цикла, найти сумму элементов, указанных в конкретном варианте. **Составить блок схему алгоритма и программу**.

1.3 Варианты

1) Найти сумму целых положительных чисел, кратных 3 и меньших 200.

2) Найти сумму целых положительных четных чисел, меньших 100.

3) Найти сумму целых положительных нечетных чисел, меньших 200.

4) Найти сумму целых положительных чисел, больших 20, меньших 100 и кратных 3

5) Найти сумму ряда с точностью ε=10-4, общий член которого

$$a_n = \frac{(-1)^{n-1}}{n^n}$$

6) Найти сумму ряда с точностью ε=10-4, общий член которого

$$a_n = \frac{1}{2^n} + \frac{1}{3^n}$$

7) Найти сумму ряда с точностью ε=10-4, общий член которого

$$a_n = \frac{1}{((3n-2)(3n+1))}$$

8) Найти сумму ряда с точностью ε=10-4, общий член которого

$$a_n = \frac{(2n-1)}{2^n}$$

9) Найти сумму ряда с точностью ε=10-4, общий член которого

$$a_n = \frac{10^n}{n!}$$

10) Найти сумму ряда с точностью ε=10-4, общий член которого

$$a_n = \frac{n!}{(2n)!}$$

11) Найти сумму ряда с точностью ε=10-4, общий член которого

$$a_n = \frac{n!}{n^n}$$

12) Найти сумму ряда с точностью ε=10-4, общий член которого

$$a_n = \frac{2^n n!}{n^n}$$

13) Найти сумму ряда с точностью ε=10-4, общий член которого

$$a_n = \frac{3^n n!}{(3n)!}$$

14) Найти сумму ряда с точностью ε=10-4, общий член которого

$$a_n = \frac{n!}{3n^n}$$

15) Найти сумму ряда с точностью ε=10-4, общий член которого

$$a_n = \frac{(n!)^2}{2^{n^2}}$$

16) Найти сумму ряда с точностью ε=10-4, общий член которого

$$a_n = \lg(n!)e^{-n\sqrt{n}}$$

17) Найти сумму ряда с точностью ε=10-4, общий член которого

$$a_n = 10^{-n}(n-1)!$$

18) Найти сумму ряда с точностью ε=10-4, общий член которого

$$a_n = \frac{n^3}{(3n-3)!}$$

19) Найти сумму ряда с точностью ε=10-4, общий член которого

$$a_n = \frac{n}{(n-1)^2}$$

20) Найти сумму ряда с точностью ε=10-4, общий член которого

$$a_n = e^n \cdot 100^{-n^2}$$

21) Найти сумму 13 членов ряда, в котором

$$a_n = \frac{\ln(n!)}{n^2}$$

22) Найти сумму 15 членов ряда, в котором

$$a_n = \frac{n^{\ln n}}{(\ln n)^n}$$

23) Найти сумму 10 членов ряда, в котором

$$a_n = \frac{n!}{n^{\sqrt{n}}}$$

24) Найти сумму 9 членов ряда, в котором

$$a_n = e^{-\sqrt{n}}$$

25) Найти сумму 7 членов ряда, в котором

$$a_n = n^2 e^{-\sqrt{n}}$$

1.4. Методические указания

1. При определении суммы членов ряда следует использовать рекуррентную формулу для получения следующего члена ряда.

Например, требуется найти сумму ряда с точностью ε=10-4, общий член которого $a_n = \frac{2(n!)^2}{(3(2n)!)!}$.

Для получения рекуррентной формулы вычислим отношение:

$$\frac{a_{n+1}}{a_n} = \frac{2((n+1)!)^2 \cdot 3(2n)!}{3(2n+2)! \cdot 2(n!)^2} = \frac{n+1}{2(2n+1)},$$

откуда:

$$a_{n+1} = a_n \cdot \frac{(n+1)}{2(2n+1)}.$$

1. При составлении программы считать, что точность достигнута, если an <ε

Задание 2. Преобразование символьных строки.

Цель: Изучение символьных и строковых переменных и способов их обработки в языке С++.

2.1. Краткие теоретические сведения

Для представления символьной (текстовой) информации можно использовать символы, символьные переменные и символьные константы.

Символьная константа представляется последовательностью символов, заключенной в кавычки: “Начало строки \n”. В Си нет отдельного типа для строк. Массив символов - это и есть строка. Количества элементов в таком массиве на один элемент больше, чем изображение строки, т. к. в конец строки добавлен ‘\0’ (нулевой байт или нуль-терминатор).

A	A | \0
‘A’	“A”
символ(1 байт)	строка (2 байта)

Присвоить значение массиву символов с помощью обычного оператора присваивания нельзя. Поместить строку в массив можно либо при вводе, либо с помощью инициализации:

char s[] = “ABCDEF”;

Для работы со строками существует специальная библиотека string.h. Примеры функций для работы со строками из библиотеки string.h:

Функция	Прототип и краткое описание функции
strcmp	int strcmp(const char *str1, const char *str2); Сравнивает строки str1 и str2. Если str1< str2, то результат отрицательный, если str1 = str2, то результат равен 0, если str1> str2, то результат положительный.
strcpy	char* strcpy(char*s1, const char *s2); Копирует байты из строки s1 в строку s2
strdup	char *strdup (const char *str); Выделяет память и перености в нее копию строки str.
strlen	unsigned strlen (const char *str); Вычисляет длину строки str.
strncat	char *strncat(char *s1, const char *s2, int kol); Приписывает kol символов строки s1 к строке s2.
strncpy	char *strncpy(char *s1, const char *s2, int kol); Копирует kol символов строки s1 в строку s2.
strnset	char *strnset(char *str, int c, int kol); Заменяет первые kol символов строки s1 символом c.

Строки, при передаче в функцию, в качестве фактических параметров могут быть определены либо как одномерные массивы типа char[], либо как указатели типа char*. В отличие от обычных массивов в этом случае нет необходимости явно указывать длину строки.

2.2. Постановка задачи

Задана строка, состоящая из символов. Символы объединяются в слова. Слова друг от друга отделяются одним или несколькими пробелами. В конце текста ставится точка. Текст содержит не более 255 символов. **Составить блок схему алгоритма и программу**, которая выполнит ввод строки, используя функцию gets(s) и осуществит обработку строки в соответствии со своим вариантом.

2.3. Варианты

1. Проверить является ли строка палиндромом. (Палиндром - это выражение, которое читается одинакова слева направо и справа налево).
2. Напечатать самое длинное и самое короткое слово в этой строке.
3. Напечатать все слова, которые не содержат гласных букв.
4. Напечатать все слова, которые содержат по одной цифре.
5. Напечатать все слова, которые совпадают с ее первым словом.
6. Преобразовать строку таким образом, чтобы сначала в ней были напечатаны только буквы, а потом только цифры, не меняя порядка следования символов в строке.
7. Преобразовать строку так, чтобы все буквы в ней были отсортированы по возрастанию.
8. Преобразовать строку так, чтобы все цифры в ней были отсортированы по убыванию.
9. Преобразовать строку так, чтобы все слова в ней стали идентификаторами, слова состоящие только из цифр - удалить.
10. Напечатать все слова-палиндромы, которые есть в этой строке(см 1 вариант).
11. Преобразовать строку таким образом, чтобы в ее начале были записаны слова, содержащие только цифры, потом слова, содержащие только буквы, а затем слова, которые содержат и буквы и цифры.

12. Преобразовать строку таким образом, чтобы все слова в ней были напечатаны наоборот.

13. Преобразовать строку таким образом, чтобы буквы каждого слова в ней были отсортированы по возрастанию.

14. Преобразовать строку таким образом, чтобы цифры каждого слова в ней были отсортированы по убыванию.

15. Преобразовать строку таким образом, чтобы в ней остались только слова, содержащие буквы и цифры, остальные слова удалить.

16. Определить какое слово встречается в строке чаще всего.

17. Определить какие слова встречаются в строке по одному разу.

18. Все слова строки, которые начинаются с буквы, отсортировать в алфавитном порядке.

19. Все слова строки, которые начинаются с цифры отсортировать по убыванию.

20. Удалить из строки все слова, которые не являются идентификаторами.

21. Определить сколько раз в строке встречается заданные буквы.

22. Определить сколько раз в строке встречаются заданные цифры.

23. Определить сколько раз в строке встречаются заданные слова.

24. Перекодировать строку так, что бы выходная строка содержала символы, код которых был дополнением до числа 256 (новый код = 256 - старый код).

25. Перекодировать строку так, что бы выходная строка содержала символы, код которых получался из исходных путем инвертирования всех бит.

Задание 3 Строковый ввод-вывод

Цель: Работа с текстовыми файлами, ввод-вывод текстовой информации и ее хранение на внешних носителях.

3.1 Краткие теоретические сведения

Для построчного ввода - вывода используются следующие функции;

1) char *fgets(char *s, int n, FILE *F), где
char *s - адрес, по которому размещаются считанные байты;
int n - количество считываемых байтов;
FILE *fp - указатель на файл, из которого производится считывание.
Прием символов заканчивается после передачи n байтов или при получении "\n". Управляющий символ "\n" тоже передается в принимающую строку. В любом случае строка заканчивается "\0". При успешном завершении считывания, функция возвращает указатель на прочитанную строку, иначе возвращает NULL.
2) char *fputs(char *s, FILE *F), где
char *s - адрес, из которого берутся записываемые в файл байты;
FILE *fp - указатель на файл, в который производится запись.
Пример:

```
int MAXLINE=255; //максимальная длина строки
FILE    *in, //исходный файл
        *out; //принимающий файл
char buf[MAXLINE];    //строка, с помощью которой выполняется копирование
//копирование строк одного файла в другой
while (fgets (buf, MAXLINE, in)!=NULL)
fputs(buf,out);
```

3.2. Постановка задачи

Задан текстовый файл F1, содержащий не менее, чем 100 текстовых строк (количество строк в файле заранее не известно).
Составить блок схему алгоритма и программу согласно варианта задания.

3.3. Варианты

1. Скопировать в файл F2 только четные строки из F1.

2.Скопировать в файл F2 только те строки из F1, которые начинаются с буквы «А».

3. Скопировать в файл F2 только те строки из F1, которые начинаются и заканчиваются на одну и ту же букву.

4.Скопировать из файла F1 в файл F2 строки, начиная с 4.

5.Скопировать из файла F1 в файл F2 строки, начиная с K до K+5.

6.Скопировать из файла F1 в файл F2 строки, начиная с N до K.

7. Скопировать из файла F1 в файл F2 все строки, кроме тех, что начинаются на букву A.

8.Скопировать из файла F1 в файл F2 все строки, которые не содержат цифры.

9. Скопировать из файла F1 в файл F2 все строки, которые содержат только одно слово.

10.Скопировать из файла F1 в файл F2 все строки, которые не содержат слова , начинающиеся на одну букву.

11.Скопировать из файла F1 в файл F2 все строки, кроме той строки, которая содержит самое короткое слово.

12.Скопировать из файла F1 в файл F2 все строки, кроме той строки, в которой больше всего гласных букв.

13. Скопировать из файла F1 в файл F2 все строки, начинающиеся на букву «A» и расположенные между строками с номерами N1 и N2.

14. Скопировать из файла F1 в файл F2 все строки, не содержащие букву «A» и расположенные между строками с номерами N1 и N2.

15.

1) Скопировать из файла F1 в файл F2 все строки, заканчивающиеся на букву «A» и расположенные между строками с номерами N1 и N2.

16. Скопировать из файла F1 в файл F2 все строки, начинающиеся на букву «A» и Заканчивающиеся на букву «C», расположенные между строками с номерами N1 и N2.

17. Скопировать из файла F1 в файл F2 все строки, начинающиеся на букву «A» расположенные между строками с номерами N1 и N2, а затем все строки от N2+3 и до последней.

18. Скопировать из файла F1 в файл F2 все строки, в которых нет одинаковых слов.

19. Скопировать из файла F1 в файл F2 все строки, в которых нет слов, совпадающих с первым словом.

20. Скопировать из файла F1 в файл F2 все строки, в которых есть одинаковые слова.

21. Скопировать из файла F1 в файл F2 все строки, в которых есть слова, совпадающие с первым словом.

22. Скопировать из файла F1 в файл F2 все строки, в которых более 2 слов.

23. Скопировать из файла F1 в файл F2 все строки, в которых содержится только одно слово.

24. Скопировать из файла F1 в файл F2 все строки, в которых содержится два одинаковых слова.

25. Скопировать из файла F1 в файл F2 все строки, в которых содержится не менее двух одинаковых слов.

Список рекомендуемых источников

1. Подбельский В. В., Фомин С. С. Программирование на языке Си: Учеб. пособие. –М.:Финансы и статистика, 1998.–600с.
2. Подбельский В. В. Язык Си++: Учеб. пособие.–М.:Финансы и статистика, 1996.–560с.
3. Страуструп Б. Язык программирования Си++: Пер. с англ. – М.: Радио и связь, 1991.-352с.

Приложение 1

Класичний приватний університет

Кафедра програмування та інформаційних технологій

“Отримано”
Реєстраційний номер № _____
від “____” _____ 200__ р.

“Відправлено з зауваженнями”
Реєстраційний номер № _____
від “____” _____ 200__ р.

“Отримано повторно”
Реєстраційний номер № _____
від “____” _____ 200__ р.

КОНТРОЛЬНА РОБОТА № _____

з дисципліни _____
на тему

_____

_____

Виконав (ла)
студент (ка) ____ курсу, групи _____

_____
(прізвище, ім’я, по батькові)

Перевірив:

_____
(оцінка, дата, підпис викладача) (прізвище, ім’я, по батькові викладача)
__

Адреса університету:

69002, м.Запоріжжя,
вул. Жуковського, 70-б.

тел. 63-99-73

Адреса для повідомлення результату контрольної роботи студента:

_____

_____
(індекс, населенний пункт, вулиця, будинок, квартира)

телефон: _____

м. Запоріжжя, 200___ р.