МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ

по лабораторной работе №1
по дисциплине «Алгоритмы и структуры данных»
Тема: Рекурсия

Студент гр. 7382 _____ **Глазунов С.А.**

Преподаватель _____ **Фирсов М. А.**

Санкт-Петербург
2018

Цель работы.

Ознакомиться с основными методами использования рекурсии и написать программу с использованием рекурсии.

Основные теоретические положения.

Реку́рсия — определение, описание, изображение какого-либо объекта или процесса внутри самого этого объекта или процесса, то есть ситуация, когда объект является частью самого себя.

Ход работы:

В лабораторной работе требуется написать программу, которая по заданному константному_выражению вычисляет его значение либо сообщает о переполнении (превышении заданного значения) в процессе вычислений.:

В работе используется язык программирования C++;

исходный код: файл main.c

Создается функция recmath() , которая выполняет поставленную перед нами задачу; в программе есть переменная pad, которая указывает на глубину рекурсии; с её помощью на экран выводиться наглядный вызов каждой функции с отступами (чем правее тем глубже глубина рекурсии).

В функции main() используется функция textcorrect, которая проверяет входные данные на корректность. Программа завершается когда закончится строка с примером (тестирование происходит из файла) . Программа может менять свое поведение в зависимости от включенных препроцессоров.

Также чтобы сохранить приоритеты операция,а именно сначала идет «*»,а только потом «+» была написана функция multiply(),которая и позволяет считать произведение ,когда рекурсия еще идет в глубину, соответственно,что при обратном ходе идет сложение и сохраняется порядок операций.

Файл компиляции и запуска программы из файла на тестирование — runtest.sh:

Компилирует программу лежащую в папке Sourse и запускает её подавая данные на вход программы файлы из папки: tests.

Программа может как и печатать ход работы алгоритма,так и просто выводить решение примера на консоль, используя возможности препроцессора.

Тестирование программы:

Программа выдает ошибку «error», если на вход подается строка ,где есть символы;где первый или последний символ является знаком операции;где присуща неправильная запись примера (5+*5-неправильно);также она выдает ошибку ,если вводимые числа слишком большие,то есть превышают по модулю INT_MAX. Тестирование программы осуществляется скриптом написанным на bashe в файле runtest.sh.

Выводы.

В ходе выполнения лабораторной работы получены знания по теме «рекурсия» и закрепленны знания синтаксиса языка C++;

ИСХОДНЫЙ КОД:

```
#INCLUDE<IOSTREAM>
#INCLUDE<FSTREAM>
#INCLUDE<CLIMITS>//FOR INT_MAX AND INT_MIN
#INCLUDE<CSTDLIB>//FOR EXIT
#INCLUDE<CMATH>//FOR POW
INT STRTOINT(STD::STRING TEXT,INT SIZE);
//#DEFINE TEST          //FOR ILLUSTATION OF RECURSION
#DEFINE SCRIPT_TEST   //FOR RUNTEST.SH
BOOL IS_NUMBER(STD::STRING TEXT)
{
        INT MAX_VALUE[10]={2,1,4,7,4,8,3,6,4,7};
        IF((TEXT[0]!='-'&&TEXT.SIZE()<10)||(TEXT[0]=='-'&&TEXT.SIZE()<11))
                RETURN TRUE;


        INT I=0;
        INT J=0;
        IF(TEXT[0]=='-')
                I=1;
        INT HIGHPOINT=10+I;
        FOR(;I<HIGHPOINT;I++)
        {
                IF(TEXT[I]>MAX_VALUE[J++])
                        RETURN FALSE;
        }


        RETURN TRUE;
}
VOID FIND_SIZE_OF_INT(STD::STRING TEXT,INT &I,INT &SIZE)
{
WHILE((TEXT[I]<='9'&&TEXT[I]>='0')||TEXT[I]=='-')//IT'S A SEARCH OF NUMBER IN
STRING
  {
    I++;
    SIZE++;
  }
}
```

```
INT MULTIPLY(STD::STRING TEXT,INT &SHIFT,INT PAD,INT BONUS_PAD)
{
   INT I=SHIFT;
   INT SIZE=0;
   INT TOTAL;
   STD::STRING CUR;

#IFDEF TEST                                    //THIS PART FOR ILLUSTRATION OF RECURSION
   FOR(INT K=0; K<PAD; K++)                        //
      STD::COUT<<"\T";                          //
                FOR(INT K=0; K<BONUS_PAD; K++)                  //
                STD::COUT<<" ";                           //
                        STD::COUT<<PAD<<"."<<BONUS_PAD<<".MULTIPLY"<<STD::ENDL;
//
#ENDIF

   FIND_SIZE_OF_INT(TEXT,I,SIZE);//IT'S A SEARCH OF NUMBER IN STRING
   CUR=TEXT.SUBSTR(SHIFT,SIZE);//TAKE A PART OF STRING FOR TRANSMUTATION
IN INT
   SHIFT=I;
   TOTAL=STRTOINT(CUR,SIZE);

   IF(TEXT[SHIFT]=='*')
   {
      SHIFT++;
      RETURN TOTAL*MULTIPLY(TEXT,SHIFT,PAD,++BONUS_PAD);
   }

   RETURN TOTAL;
}

INT RECMATH(STD::STRING TEXT,INT SHIFT,INT PAD) {//THE FIRST RECURSION
   IF(TEXT.SIZE()==SHIFT)//EXIT OF RECURSION
      RETURN 0;//

   INT SIZE=0,TOTAL,I=SHIFT;//TOTAL-IS ANSWER;
```

```
   STD::STRING CUR;

   FIND_SIZE_OF_INT(TEXT,I,SIZE);
   CUR=TEXT.SUBSTR(SHIFT,SIZE);//TAKE A PART OF STRING FOR TRANSMUTATION
IN INT
   SHIFT=I;
   TOTAL=STRTOINT(CUR,SIZE);


#IFDEF TEST
   FOR(INT K=0; K<PAD; K++)
      STD::COUT<<"\T";
   STD::COUT<<PAD<<".RECMATH"<<STD::ENDL;
#ENDIF
   IF(TEXT[SHIFT]=='+'){
      INT CURCHECK=RECMATH(TEXT,SHIFT+1,++PAD);


      IF(TOTAL>INT_MAX-CURCHECK)
{
#IFDEF SCRIPT_TEST
   FILE *F;
   F = FOPEN("OUTPUT_OF_LAB1.TXT","WT");
   FPRINTF(F, "ERROR" );
   FCLOSE(F);
#ELSE
   STD::COUT<<"ERROR"<<STD::ENDL;
      EXIT(0);
#ENDIF
}

   RETURN TOTAL+CURCHECK;
}

   IF(TEXT[SHIFT]=='*')//THE SECOND RECURSION(MULTIPLY)
   {
       SHIFT++;
       INT CURCHECK=MULTIPLY(TEXT,SHIFT,PAD,0);
       IF(CURCHECK!=0)
```

```
        IF(ABS(TOTAL)>INT_MAX/ABS(CURCHECK))

                {
                        #IFDEF SCRIPT_TEST
                FILE *F;
                F = FOPEN("OUTPUT_OF_LAB1.TXT","WT");
                FPRINTF(F, "ERROR" );
                FCLOSE(F);
                EXIT(0);
                        #ELSE
                STD::COUT<<"ERROR"<<STD::ENDL;
          EXIT(0);
                        #ENDIF
                }

      TOTAL*=CURCHECK;
      RETURN TOTAL+RECMATH(TEXT,SHIFT,++PAD);//
    }

    RETURN TOTAL;
}

INT STRTOINT(STD::STRING TEXT,INT SIZE) {       //THIS FUNCTION IS LIKE ATOI IN C
    INT TOTAL=0;
    INT I=0;
    INT CHECK=1;
    IF(TEXT[0]=='-') {
      I=1;
      CHECK=-1;
    }

    FOR(; I<SIZE; I++)
    {
      TOTAL+=((INT)(TEXT[I])-'0')*POW(10,SIZE-I-1);
    }
```

```
  IF(!IS_NUMBER(TEXT))//COMPARE SIZE OF NUMBER WITH NORMAL SITUATION
                {
                #IFDEF SCRIPT_TEST
                  FILE *F;
                F = FOPEN("OUTPUT_OF_LAB1.TXT","WT");
                FPRINTF(F, "ERROR" );
                FCLOSE(F);
                EXIT(0);
                #ELSE
                STD::COUT<<"ERROR"<<STD::ENDL;
        EXIT(0);
                #ENDIF
                }

  RETURN CHECK*TOTAL;
}

BOOL TEXTCORRECT(STD::STRING TEXT){                  //THIS FUNCTION CHECKS
STDIN
  INT CHECK=0;
  IF(TEXT[TEXT.SIZE()-1]<'0'||TEXT[TEXT.SIZE()-1]>'9')// IF STDIN IS WRONG FOR
CALCULATOR
    RETURN FALSE;//CHECK LAST SYMBOL              //FUNC. SEND FALSE

  FOR(INT I=0;I<TEXT.SIZE();I++)
  {
    IF(ISALPHA(TEXT[I])||TEXT[I]=='.'||TEXT[I]==',')//CHECK SYMBOL IN STRING
      RETURN FALSE;
  }

  IF(TEXT[0]<'0'||TEXT[0]>'9')//CHECK THE FIRST SYMBOL
        IF(TEXT[0]!='-')
          RETURN FALSE;

  FOR(INT I=0;I<TEXT.SIZE();I++)// IF THERE IS A SITUATION LIKE "5+*3" -IT'S FALSE
  {
    IF(CHECK==2)
```

```
            RETURN FALSE;
        IF(TEXT[I]=='+'||TEXT[I]=='*')
        {
            CHECK++;
            CONTINUE;
        }
        CHECK=0;

    }

RETURN TRUE;
}

INT MAIN()
{
    STD::STRING TEXT;
    INT FIN;
#IFNDEF SCRIPT_TEST
    STD::COUT<<"HELLO,I AM CALCULATOR.ENTER..."<<STD::ENDL;
#ENDIF
    GETLINE(STD::CIN,TEXT);
    FOR(INT I=0;I<TEXT.SIZE();I++)
                IF(TEXT[I]==' ')
        {                           // "THIS IS AN EXAMPLE SENTENCE."
        TEXT.ERASE (I,1);
        I--;
        }
    BOOL ISCORRECT=TEXTCORRECT(TEXT);

    IF(ISCORRECT)
    FIN=RECMATH(TEXT,0,0);
#IFDEF SCRIPT_TEST
    FILE *F;
    F = FOPEN("OUTPUT_OF_LAB1.TXT","WT");
    IF(ISCORRECT){
    FPRINTF(F, "%D",FIN);
    }
```

```
   ELSE
   {
      FPRINTF(F, "ERROR" );
   }
   FCLOSE(F);
#ELSE
   IF(!ISCORRECT){
      STD::COUT<<"ERROR"<<STD::ENDL;
      RETURN 0;
   }
   STD::COUT<<"RESULT="<<FIN<<STD::ENDL;
#ENDIF
   RETURN 0;
}
```