|  |  |
| --- | --- |
|  | **Отчет по лабораторной работе № 24**  Студент группы М8О-103Б-23 Махмутов Дэниз Ирикович, № по списку 17  Контакты www, e-mail, icq, skype lustmodesta@gmail.com  Работа выполнена: « 8» апреля 2024 г.  Преподаватель: доцент каф. 806 Никулин С.П.  Входной контроль знаний с оценкой  Отчет сдан « » 2024 г., итоговая оценка  Подпись преподавателя |

1. **Тема:** Программирование на интерпретируемых командных языках
2. **Цель работы:** составить программу выполнения заданных действий над файлами на одном из интерпретируемых командных языков ОС UNIX

3**. Задание (***вариант №* 12 **):** вынести общие сомножители (переменные и константы) из разности. (29-17)

**Оборудование** (*лабораторное*):

Оборудование (лабораторное):

ЭВМ , процессор , имя узла сети c ОП Мб,

НМД Мб. Терминал адрес . Принтер

Другие устройства

Оборудование ПЭВМ студента, если использовалось:

Процессор AMD Ryzen 5 5600x, 6 Cores с ОП Ubuntu 22.04.3 LTS Мб,

НМД 486,400 Мб. Монитор

Другие устройства

1. **Программное обеспечение** (*лабораторное*):

Операционная система семейства , наименование версия

Интерпретатор команд bash версия

Система программирования gcc версия

Редактор текстов версия

Утилиты операционной системы

Прикладные системы и программы

Местонахождения и имена файлов программ и данных

*Программное обеспечение ЭВМ студента, если использовалось:*

*Операционная система семейства Linux, наименование Ubuntu версия 22.04.3 LTS*

*интерпретатор команд GNU bash версия 5.1.16(1)-release (x86/64-pc-linux-gnu)*

*Система программирования C версия 2.35*

*Редактор текстов GNU emacs версия 27.1*

1. **Идея, метод, алгоритм** решения задачи (в формах: словесной, псевдокода, графической [блок-схема, диаграмма, рисунок, таблица] или формальное описание с пред- и постусловиями)

Программа получает строку без пробелов. Пользователь использует круглые скобки для задания арифметических выражений. Количество скобок должно быть вдвое больше количества арифметических выражений.

Сначала создается бинарное дерево из введенной строки. В дереве каждый узел представляет арифметическое действие или число.

Затем программа преобразует дерево. Если она находит узел с арифметическим действием "\*", у которого потомки это "-", она заменяет этот узел на "\*", а значение правого потомка становится "-", а потомки становятся множителями.

1. **Сценарий выполнения работы** [план работы, первоначальный текст программы в черновике (можно на отдельном листе) и тесты, либо соображения по тестированию].

parstree определяет узел бинарного дерева, хранящего символьное значение (арифметический оператор или операнд) и указатели на левого, правого потомка и родителя.

Функция printpt выводит бинарное дерево в виде арифметического выражения.

Функция printt выводит структуру дерева с отступами для визуализации.

Функция rbuildt преобразует дерево, вынося общий множитель из суммы или разности.

Функция getptree строит бинарное дерево разбора арифметического выражения из строки.

В main пользователь вводит арифметическое выражение, которое преобразуется в бинарное дерево, затем дерево преобразуется функцией rbuildt, и в итоге выводится измененное арифметическое выражение.

*Пункты 1-7 отчёта составляются* ***строго до*** *начала лабораторной работы.*

Допущен к выполнению работы. Подпись преподавателя

1. **Распечатка протокола** (подклеить листинг окончательного варианта программы с текстовыми примерами, подписанный преподавателем)

|  |
| --- |
| unix@DESKTOP-MPQDBS2:~/labs//24$ cat 24.c  #include<stdio.h>  #include<stdlib.h>  #include<malloc.h>  struct parstree{  char value;  struct parstree\* left;  struct parstree\* right;  struct parstree\* prev;  };  typedef struct parstree PTREE;  void printpt(PTREE\* root){  if(root->left != 0){  printf("(");  printpt(root->left);  }  printf("%c",root->value);  if(root->right != 0){  printpt(root->right);  printf(")");  }  }  void printt(PTREE\* root, int counter){  if(root->left !=0)printt(root->left, counter + 1);  for(int i = 0;i<counter;i++){  printf(" ");  }  printf("\_%c\n",root->value);  if(root->right != 0)printt(root->right, counter + 1);  }  PTREE\* rbuildt(PTREE\* root){  PTREE\* temp;  temp = root;  char general;  char tempc;  if(temp->value == '-')  if(temp->left != 0 && temp->right != 0 && temp->left->value == temp->right->value && temp->left->value == '\*')  if(temp->left->left->left == 0 && temp->left->left->right ==0 && temp->right->left->left == 0 && temp->right->left->right == 0  && temp->right->right->left == 0 && temp->right->right->right == 0 && temp->left->right->left == 0 && temp->left->right->right == 0){  if(temp->left->left->value == temp->right->left->value){  general = temp->left->left->value;  tempc = temp->value;  temp->value = temp->right->value;  temp->right->value = tempc;  temp->left->value = general;//  temp->right->left->value = temp->left->right->value;  free(temp->left->left);  temp->left->left = 0;  free(temp->left->right);  temp->left->right = 0;  return (temp);  }  if(temp->left->left->value == temp->right->right->value){  general = temp->left->left->value;  tempc = temp->value;  temp->value = temp->right->value;  temp->right->value = tempc;  temp->left->value = general;//  temp->right->right->value = temp->right->left->value;  temp->right->left->value = temp->left->right->value;  free(temp->left->left);  temp->left->left = 0;  free(temp->left->right);  temp->left->right = 0;  return (temp);  }  if(temp->left->right->value == temp->right->left->value){  general = temp->left->right->value;  tempc = temp->value;  temp->value = temp->right->value;  temp->right->value = tempc;  temp->left->value = general;//  temp->right->left->value = temp->left->left->value;  free(temp->left->left);  temp->left->left = 0;  free(temp->left->right);  temp->left->right = 0;  return (temp);  }  if(temp->left->right->value == temp->right->right->value){  general = temp->left->right->value;  tempc = temp->value;  temp->value = temp->right->value;  temp->right->value = tempc;  temp->left->value = general;//  temp->right->right->value = temp->right->left->value;  temp->right->left->value = temp->left->left->value;  free(temp->left->left);  temp->left->left = 0;  free(temp->left->right);  temp->left->right = 0;  return (temp);  }  }  if(temp->left != 0) temp->left = rbuildt(temp->left);  if(temp->right != 0) temp->right = rbuildt(temp->right);  return (temp);  }  PTREE\* getptree(PTREE\* root, char\* c, int len){  PTREE \* temp;  temp = root;  for(int i= 0; i < len-1;i++){  if(c[i] == '('){  temp->left = malloc(sizeof(PTREE));  temp->left->prev = temp;  temp= temp->left;  temp->left = 0;  temp->right = 0;  continue;  }  if(c[i] == ')'){  temp= temp->prev;  continue;  }  if(c[i] == '+' || c[i] == '-' || c[i] == '/' || c[i] == '\*' || c[i] == '^'){  temp->value = c[i];  temp->right = malloc(sizeof(PTREE));  temp->right->prev = temp;  temp = temp->right;  temp->left = 0;  temp->right = 0;  continue;  }  else{  temp->value = c[i];  temp = temp->prev;  continue;  }  }  return (temp);  }  int main(){  char\* c;  c = malloc(sizeof(char));  char temp;  int len = 0;  printf("Enter arithmetical expression: ");  while(1){  scanf("%c", &temp);  if(temp == '\n') break;  len++;  c = realloc(c, len\*sizeof(char));  c[len-1] = temp;  }  PTREE\* root;  root = malloc(sizeof(PTREE));  root->prev = 0;  root->left = 0;  root->right = 0;  printf("[%s], %d\n", c, len);  printf("Creating of tree...\n");  root = getptree(root, c, len);  printf("Created parsing tree: \n");  printt(root, 0);  root = rbuildt(root);  printf("\n");  printf("Changed Tree: \n");  printt(root, 0);  // printf("|%c",root->value);  printf("Changed expression: ");  printpt(root);  printf("\n");  return 0;  }  unix@DESKTOP-MPQDBS2:~/labs//24$ cc -o 24.out 24.c  unix@DESKTOP-MPQDBS2:~/labs//24$ ./24.out  Enter arithmetical expression: (1+2)  [(1+2)], 5  Creating of tree...  Created parsing tree:  \_1  \_+  \_2  Changed Tree:  \_1  \_+  \_2  Changed expression: (1+2)  unix@DESKTOP-MPQDBS2:~/labs//24$ ./24.out  Enter arithmetical expression: ((1+2)\*3)  [((1+2)\*3)], 9  Creating of tree...  Created parsing tree:  \_1  \_+  \_2  \_\*  \_3  Changed Tree:  \_1  \_+  \_2  \_\*  \_3  Changed expression: ((1+2)\*3)  unix@DESKTOP-MPQDBS2:~/labs//24$ ./24.out  Enter arithmetical expression: ((1\*3)-(4\*3))  [((1\*3)-(4\*3))], 13  Creating of tree...  Created parsing tree:  \_1  \_\*  \_3  \_-  \_4  \_\*  \_3  Changed Tree:  \_3  \_\*  \_1  \_-  \_4  Changed expression: (3\*(1-4))  unix@DESKTOP-MPQDBS2:~/labs//24$ ./a.out  bash: ./a.out: No such file or directory  unix@DESKTOP-MPQDBS2:~/labs//24$ ./24.out  Enter arithmetical expression: (1\*((3\*4)-(4\*a)))  [(1\*((3\*4)-(4\*a)))], 17  Creating of tree...  Created parsing tree:  \_1  \_\*  \_3  \_\*  \_4  \_-  \_4  \_\*  \_a  Changed Tree:  \_1  \_\*  \_4  \_\*  \_3  \_-  \_a  Changed expression: (1\*(4\*(3-a)))  unix@DESKTOP-MPQDBS2:~/labs//24$ ./24.out  Enter arithmetical expression: ((((3\*a)-(b\*3)^3)\*5)-6)  [((((3\*a)-(b\*3)^3)\*5)-6)], 23  Creating of tree...  Created parsing tree:  \_3  \_\*  \_a  \_^  \_3  \_\*  \_5  \_-  \_6  Changed Tree:  \_3  \_\*  \_a  \_^  \_3  \_\*  \_5  \_-  \_6  Changed expression: ((((3\*a)^3)\*5)-6) |

1. **Дневник отладки** должен содержать дату и время сеансов отладки, и основные ошибки (ошибки в сценарии и программе, не стандартные операции) и краткие комментарии к ним. В дневнике отладки приводятся сведения об использовании других ЭВМ, существенном участии преподавателя и других лиц в написании и отладке программы.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Лаб.  или  дом. | Дата | Время | Событие | Действие по исправлению | Примечание |
|  |  |  |  |  |  |  |

1. Замечание автора по существу работы
2. Выводы смогли составить программу выполнения заданных действий над файлами на одном из интерпретируемых языков ОС UNIX, разобрали структуру интерпретируемых языков, способы программирования скриптов.

Недочеты, допущенные при выполнении задания, могут быть устранены следующим образом

Подпись студента