SYSTEM SOFTWARE AND OPERATING SYSTEMS LAB. (17CSL67) BLLDEACET/LM/CS/15CSL67 Page 1 1(a) Lex program to recognize a valid arithmetic expression and to recognize the identifiers and operators present. Print them separately. %{ #include int oprc=0,idc=0,digc=0,top=-1; char s[20]; %} iden [a-zA-Z]+[a-zA-Z0-9]\* opr [+-/\*] digit [0-9]+ %% [ \n\t]+ {;} ['('] { s[++top]='('; } [')'] { if(s[top]=='(' && top!=-1) top--; else { printf("\nInvalid Expression.\n"); exit(0); } } {iden} {idc++;} {digit} {digc++;} {opr} {oprc++;} . { printf("\nInvalid Expression.\n"); exit(0); } %% main() { system("clear"); printf("Enter an Arithmetic Expression:\n"); yylex(); if(((idc+digc)==oprc+1)&&top==-1) { printf("\nValid Expression.\n"); printf("\nNumber of Operators= %d\nNumber of Identifiers= %d\nNumber of Digits= %d\n",oprc,idc,digc); } else SYSTEM SOFTWARE AND OPERATING SYSTEMS LAB. (17CSL67) BLLDEACET/LM/CS/15CSL67 Page 2 printf("\nInvalid Expression.\n"); } yywrap() {} Execution: [root@cse1bldea ssw]# lex 2a.lex [root@cse1bldea ssw]# cc lex.yy.c [root@cse1bldea ssw]# ./a.out Output: I. Enter an Arithmetic Expression: ((a+b)-c\*9)/2 Valid Expression. Number of Operators= 4 Number of Identifiers= 3 Number of Digits= 2 II. Enter an Arithmetic Expression: ((a+b)-c Invalid Expression. SYSTEM SOFTWARE AND OPERATING SYSTEMS LAB. (17CSL67) BLLDEACET/LM/CS/15CSL67 Page 3 1(b) YACC program to evaluate an arithmetic expression involving operators +, -, \* and /. %{ #include #include %} %token num %% START: expr '\n' {printf("\nValue= %d\n",$$); exit(0);} | error {printf("\nInvalid Expression.\n"); exit(0);} ; expr: expr '+' term {$$=$1+$3;} | expr '-' term {$$=$1-$3;} |term ; term: term '\*' factor {$$=$1\*$3;} | term '/' factor {$$=$1/$3;} | factor ; factor: '(' expr ')' {$$=$2;} | num ; %% main() { system("clear"); printf("Enter an Arithmetic Expression to Evaluate:\n"); yyparse(); } yyerror() {} yylex() { int c; c=getchar(); if(isdigit(c)) { yylval=c-'0'; return num; } return c; } SYSTEM SOFTWARE AND OPERATING SYSTEMS LAB. (17CSL67) BLLDEACET/LM/CS/15CSL67 Page 4 Execution: [root@cse1bldea ssw]# yacc -d 5a.yacc [root@cse1bldea ssw]# cc y.tab.c [root@cse1bldea ssw]# ./a.out Output: I. Enter an Arithmetic Expression to Evaluate: ((2+3)\*5)+5 Value= 30 II. Enter an Arithmetic Expression to Evaluate: (2++3) Invalid Expression. SYSTEM SOFTWARE AND OPERATING SYSTEMS LAB. (17CSL67) BLLDEACET/LM/CS/15CSL67 Page 5 2. YACC program to recognize the grammar (anb) (where n>= 10). %{ #include #include %} %token A B NEWLINE %% START: VALID {printf("\nString is Accepted.\n"); exit(0);} | error {printf("\nString is Rejected.\n"); exit(0);} ; VALID: S NEWLINE ; S: X B ; X: A A A A A A A A A A |A X ; %% main() { system("clear"); printf("Enter the string of a and b:\n"); yyparse(); } yyerror() {} Supporting Lex Program: %{ #include #include"y.tab.h" %} %% [a] {return A;} [b] {return B;} [\n] {return NEWLINE;} %% yywrap() {} SYSTEM SOFTWARE AND OPERATING SYSTEMS LAB. (17CSL67) BLLDEACET/LM/CS/15CSL67 Page 6 Execution: [root@cse1bldea ssw]# lex 6.lex [root@cse1bldea ssw]# yacc -d 6.yacc [root@cse1bldea ssw]# cc y.tab.c lex.yy.c [root@cse1bldea ssw]# ./a.out Output: I. Enter the string of a and b: aab String is Rejected. II. Enter the string of a and b: aaaaaaaaaab String is Accepted. SYSTEM SOFTWARE AND OPERATING SYSTEMS LAB. (17CSL67) BLLDEACET/LM/CS/15CSL67 Page 7 3. Design, develop and implement YACC/C program to construct Predictive / LL(1) Parsing Table for the grammar rules: A →aBa , B →bB | ε. Use this table to parse the sentence: abba$. #include #include #include void main() { char fin[10][20],st[10][20],ft[20][20],fol[20][20]; int a=0,e,i,t,b,c,n,k,l=0,j,s,m,p; printf("enter the no. of coordinates\n"); scanf("%d",&n); printf("enter the productions in a grammar\n"); for(i=0;i64)&&(st[i][j]0) { while(st[i][j]!=st[a][0]) { a++; } b=0; while(ft[a][b]!='\0') SYSTEM SOFTWARE AND OPERATING SYSTEMS LAB. (17CSL67) BLLDEACET/LM/CS/15CSL67 Page 8 { for(m=0;m64)&&(st[k][j]<=2;p++) { fin[s][p]=st[i][p]; } t=j; for(p=3;((st[i][j]!='|')&&(st[i][j]!='\0'));p++) { SYSTEM SOFTWARE AND OPERATING SYSTEMS LAB. (17CSL67) BLLDEACET/LM/CS/15CSL67 Page 11 fin[s][p]=st[i][j]; j++; } fin[s][p]='\0'; if(st[i][t]=='@') { b=0; a=0; while(st[a][0]!=st[i][0]) { a++; } while(fol[a][b]!='\0') { printf("M[%c,%c]=%s\n",st[i][0],fol[a][b],fin[s]); b++; } } else if(!((st[i][t]>64)&&(st[i][t] #include #include int k=0,z=0,i=0,j=0,c=0; char a[16],ac[20],stk[15],act[10]; void check(); void main() { puts("GRAMMAR is E->E+E \n E->E\*E \n E->(E) \n E->id"); puts("enter input string "); gets(a); c=strlen(a); strcpy(act,"SHIFT->"); puts("stack \t input \t action"); for(k=0,i=0; j %} %token ID DIG LB RB SUB MUL DIV ADD %left SUB ADD %left DIV MUL %% START: VALID {printf("\nValid Expression.\n"); exit(0);} | error {printf("\nInvalid Expression.\n"); exit(0);} ; VALID: E ; E: E ADD T | E SUB T | T ; T: T MUL F | T DIV F | F ; F: ID | DIG | LB E RB ; %% main() { system("clear"); printf("Enter an Arithmetic Expression:\n"); yyparse(); } yyerror() {} SYSTEM SOFTWARE AND OPERATING SYSTEMS LAB. (17CSL67) BLLDEACET/LM/CS/15CSL67 Page 17 Supporting Lex Program: %{ #include"y.tab.h" %} %% [0-9]+ {return DIG;} [a-zA-Z]+[a-zA-Z0-9]\* {return ID;} [(] {return LB;} [)] {return RB;} [\-] {return SUB;} [\\*] {return MUL;} [/] {return DIV;} [+] {return ADD;} %% yywrap() {} Execution: [root@cse1bldea ssw]# lex 4a.lex [root@cse1bldea ssw]# yacc -d 4a.yacc [root@cse1bldea ssw]# cc y.tab.c lex.yy.c [root@cse1bldea ssw]# ./a.out Output: I. Enter an Arithmetic Expression: ((a+b)-c)\*d Valid Expression. II. Enter an Arithmetic Expression: ((a+b) Invalid Expression. SYSTEM SOFTWARE AND OPERATING SYSTEMS LAB. (17CSL67) BLLDEACET/LM/CS/15CSL67 Page 18 5. Design, develop and implement a C/Java program to generate the machine code using Triples for the statement A = -B \* (C +D) whose intermediate code in three-address form: T1 = -B T2 = C + D T3 = T1 \* T2 A = T3 #include #include #include char op[2],arg1[5],arg2[5],result[5]; void main() { FILE \*fp1,\*fp2; fp1=fopen("input.txt","r"); fp2=fopen("output.txt","w"); while(!feof(fp1)) { fscanf(fp1,"%s%s%s%s",result,arg1,op,arg2); if(strcmp(op,"+")==0) { fprintf(fp2,"\nMOV R0,%s",arg1); fprintf(fp2,"\nADD R0,%s",arg2); fprintf(fp2,"\nMOV %s,R0",result); } if(strcmp(op,"\*")==0) { fprintf(fp2,"\nMOV R0,%s",arg1); fprintf(fp2,"\nMUL R0,%s",arg2); fprintf(fp2,"\nMOV %s,R0",result); } if(strcmp(op,"-")==0) { fprintf(fp2,"\nMOV R0,%s",arg1); fprintf(fp2,"\nSUB R0,%s",arg2); fprintf(fp2,"\nMOV %s,R0",result); } if(strcmp(op,"/")==0) { fprintf(fp2,"\nMOV R0,%s",arg1); fprintf(fp2,"\nDIV R0,%s",arg2); fprintf(fp2,"\nMOV %s,R0",result); } if(strcmp(op,"=")==0) { SYSTEM SOFTWARE AND OPERATING SYSTEMS LAB. (17CSL67) BLLDEACET/LM/CS/15CSL67 Page 19 fprintf(fp2,"\nMOV R0,%s",arg1); fprintf(fp2,"\nMOV %s,R0",result); } } fclose(fp1); fclose(fp2); getch(); } Output: input.txt T1 -B = ? T2 C + D T3 T1 \* T2 A T3 = ? output.txt MOV R0,-B MOV T1,R0 MOV R0,C ADD R0,D MOV T2,R0 MOV R0,T1 MUL R0,T2 MOV T3,R0 MOV R0,T3 MOV A,R0 SYSTEM SOFTWARE AND OPERATING SYSTEMS LAB. (17CSL67) BLLDEACET/LM/CS/15CSL67 Page 20 6(a) Lex program to count the numbers of comment lines in a given C program. Also eliminate them and copy the resulting program into separate file. %{ #include int i=0; %} id [a-zA-Z0-9 \n\t]\* %% "/\*"{id}"\*/" {i++;} %% main() { system("clear"); FILE \*fp; fp=fopen("abc.c","r"); yyin=fp; yylex(); printf("Number of Comment Lines= %d\n",i); } yywrap() {} Supporting abc.c file: /\*This is a C Program\*/ main() { int x=4; /\*Declaration of x\*/ int y=7,z; /\*Declaration of y and z\*/ z=x+y; printf("ans= %d",z); } Execution: [root@cse1bldea ssw]# lex 1b.lex [root@cse1bldea ssw]# cc lex.yy.c [root@cse1bldea ssw]# ./a.out SYSTEM SOFTWARE AND OPERATING SYSTEMS LAB. (17CSL67) BLLDEACET/LM/CS/15CSL67 Page 21 Output: main() { int x=4; int y=7,z; z=x+y; printf("ans= %d",z); } Number of Comment Lines= 3 SYSTEM SOFTWARE AND OPERATING SYSTEMS LAB. (17CSL67) BLLDEACET/LM/CS/15CSL67 Page 22 6(b) Write YACC program to recognize valid identifier, operators and keywords in the given text (C program) file. Lex File %{ #include #include "y.tab.h" extern yylval; %} %% [ \t] ; [+|-|\*|/|=|] {printf("operator is %s\n",yytext);return OP;} [0-9]+ {yylval = atoi(yytext); printf("numbers is %d\n",yylval); return DIGIT;} int|char|bool|float|void|for|do|while|if|else|return|void {printf("keyword is %s\n",yytext);return KEY;} [a-zA-Z0-9]+ {printf("identifier is %s\n",yytext);return ID;} . ; %% Yacc File %{ #include #include int id=0, dig=0, key=0, op=0; %} %token DIGIT ID KEY OP %% input: DIGIT input { dig++; } | ID input { id++; } | KEY input { key++; } | OP input {op++;} | DIGIT { dig++; } | ID { id++; } | KEY { key++; } | OP { op++;} ; %% #include extern int yylex(); extern int yyparse(); extern FILE \*yyin; main() { FILE \*myfile = fopen("sam\_input.c", "r"); if (!myfile) { printf("I can't open sam\_input.c!"); return -1; } SYSTEM SOFTWARE AND OPERATING SYSTEMS LAB. (17CSL67) BLLDEACET/LM/CS/15CSL67 Page 23 yyin = myfile; do { yyparse(); } while (!feof(yyin)); printf("numbers = %d\nKeywords = %d\nIdentifiers = %d\noperators = %d\n", dig, key,id, op); } void yyerror() { printf("EEK, parse error! Message: "); exit(-1); } Output : Input file SYSTEM SOFTWARE AND OPERATING SYSTEMS LAB. (17CSL67) BLLDEACET/LM/CS/15CSL67 Page 24 SYSTEM SOFTWARE AND OPERATING SYSTEMS LAB. (17CSL67) BLLDEACET/LM/CS/15CSL67 Page 25 7. Design, develop and execute a program in C/C++ to simulate the working of Shortest Remaining time and Round Robin Scheduling algorithms. Experiment with different Quantum sizes for the Round Robin algorithm. In all cases, determine the average turn Around time. Input can be read from keyboard or from a file. #include #include void roundrobin(); void srtf(); main() { int choice; printf("Enter the choice \n"); printf(" 1. Round Robin\n 2. SRTF\n 3. Exit \n"); scanf("%d",&choice); switch(choice) { case 1: printf("Round Robin scheduling algorithm\n"); roundrobin(); break; case 2: printf("\n \n ---SHORTEST REMAINING TIME NEXT---\n \n "); srtf(); break; case 3: exit(0); } }//end of main void roundrobin() { int n,bt[10],st[10],tat[10],tq,stat=0; int i,j,k,count,sq=0,temp; float atat=0.0; printf("Enter number of processes:\n"); scanf("%d",&n); for(i=0;itq) // when service time of a process greater than time quantum then time st[i]=st[i]-tq; //quantum value subtracted from service time else if(st[i]>=0) { temp=st[i]; // temp1 stores the service time of a process st[i]=0; // making service time equals 0 } sq=sq+temp; // utilizing temp1 value to calculate turnaround time tat[i]=sq; // turn around time } //end of for if(n==count) // it indicates all processes have completed their task because the count value break; // incremented when service time equals 0 } //end of while for(i=0;is[j].bt) { temp=s[i]; s[i]=s[j]; s[j]=temp; } } //turnaround time calculation tat[0]=s[0].bt; for(i=1;i //# include # include # include # define MAX 10 struct process { char pid[4]; int maxneed[3]; int altd[3]; int need[3]; int finish; }p[MAX]; int avl[3]; char safe[10][4]; void bankers(int n); void main() { int n,i; //clrscr(); printf("How many processes?"); scanf("%d",&n); printf("Enter processes"); for(i=0;i<=m) { for(i=0;i<=avl[0] && p[i].need[1]<=avl[1] && p[i].need[2]<=avl[2]) { strcpy(safe[j],p[i].pid); j++; count++; p[i].finish=1; avl[0]+=p[i].altd[0]; avl[1]+=p[i].altd[1]; avl[2]+=p[i].altd[2]; } else { continue; } } k++; } if(count==n) { printf("\nState is safe that is:\n"); for(i=0;i #include void FIFO(char [ ],char [ ],int,int); void lru(char [ ],char [ ],int,int); void opt(char [ ],char [ ],int,int); int main() { int ch,YN=1,i,l,f; char F[10],s[25]; printf("\n\n\tEnter the no of empty frames: "); scanf("%d",&f); printf("\n\n\tEnter the length of the string: "); scanf("%d",&l); printf("\n\n\tEnter the string: "); scanf("%s",s); for(i=0;i void main() { char operator; float num1, num2, result; printf("Simulation of a Simple Calculator\n"); printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n"); printf("Enter two numbers \n"); scanf("%f %f", &num1, &num2); fflush(stdin); printf("Enter the operator [+,-,\*,/] \n"); scanf("%s", &operator); switch(operator) { case '+': result = num1 + num2; break; case '-': result = num1 - num2; break; case '\*': result = num1 \* num2; break; case '/': result = num1 / num2; break; default : printf("Error in operationn"); break; } printf("\n %5.2f %c %5.2f = %5.2f\n", num1, operator, num2, result); } Output: Simulation of a Simple Calculator \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Enter two numbers 2 3 Enter the operator [+,-,\*,/] + 2.00 + 3.00 = 5.00 SYSTEM SOFTWARE AND OPERATING SYSTEMS LAB. (17CSL67) BLLDEACET/LM/CS/15CSL67 Page 38 b) Design, develop and implement a C/C++/Java program to simulate page replacement technique #include int n,nf; int in[100]; int p[50]; int hit=0; int i,j,k; int pgfaultcnt=0; void getData() { printf("\nEnter length of page reference sequence:"); scanf("%d",&n); printf("\nEnter the page reference sequence:"); for(i=0; imax) { max=near[j]; repindex=j; } } p[repindex]=in[i]; pgfaultcnt++; dispPages(); } else printf("No page fault"); } dispPgFaultCnt(); } void lru() { initialize(); int least[50]; for(i=0; i=0; k--) { if(pg==in[k]) { least[j]=k; found=1; break; } else found=0; } if(!found) least[j]=-9999; } int min=9999; int repindex; for(j=0; j<=i; k++) if(in[i]==in[k]) sofarcnt=sofarcnt+1; usedcnt[repin]=sofarcnt; } dispPages(); } } dispPgFaultCnt(); } void secondchance() { int usedbit[50]; int victimptr=0; initialize(); SYSTEM SOFTWARE AND OPERATING SYSTEMS LAB. (17CSL67) BLLDEACET/LM/CS/15CSL67 Page 43 for(i=0; i