PRAC 11: Write a program to optimize a generated 3-address code.(Implement in Python)

|  |  |
| --- | --- |
| **Program** | #include<iostream>  #include<fstream>  #include<string.h>  using namespace std;  struct store{  char opr,opd1,opd2,result;  };  struct store input[20];  void same\_lines(int);  void up(int);  void dead\_code(int);  void display(int);  void display\_after(int);  void code\_optimization(int no\_of\_lines)  {  cout<<endl<<"Displaying before optimization:\n";  display(no\_of\_lines);  cout<<endl;  same\_lines(no\_of\_lines);  up(no\_of\_lines);  dead\_code(no\_of\_lines);  cout<<endl<<"Displaying after optimization:\n";  display\_after(no\_of\_lines);  cout<<endl;  }  void same\_lines(int no\_of\_lines)  {  char result;  int flag=0;  //Eleminating repetation  for(int i=0;i<no\_of\_lines;i++)  {  for(int j=1;j<no\_of\_lines;j++)  {  if(i==j||(input[i].opr=='x'))  {  continue;  }  elseif((input[i].opr)==(input[j].opr)&&(input[i].opd1==input[j].opd1)&&(input[i].opd2==input[i].opd2))  {  flag=1;  input[j].opr='x';  result=input[j].result;  }  if(flag==1)  {  if((input[j].opd1)==result)  {  input[j].opd1=input[i].result;  }  else if((input[j].opd2)==result)  {  input[j].opd2=input[i].result;  }  }    }  }  }  void up(int no\_of\_lines)  {  char result,temp;  int flag=0;  //Eleminating unnecessary multiplication  for(int i=0;i<no\_of\_lines;i++)  {    if(input[i].opr=='x')  {  continue;  }  else if((input[i].opr)==('\*'))  {  if((input[i].opd1==('1'))||(input[i].opd2==('1')))  {  flag=1;  input[i].opr='x';  if((input[i].opd1==('1')))  {  temp=input[i].opd2;  result=input[i].result;    }  else  {  temp=input[i].opd1;  result=input[i].result;  }  }    for(int j=i;j<no\_of\_lines;j++)  {  if((input[j].opd1==(result)))  {  input[j].opd1=temp;  }  else if(input[j].opd2==(result))  {  input[j].opd2==temp;  }    }  }  if((input[i].opr)==('+'))  {  if((input[i].opd1==('0'))||(input[i].opd2==('0')))  {  flag=1;  input[i].opr='x';  if((input[i].opd1==('0')))  {  temp=input[i].opd2;  result=input[i].result;    }  else  {  temp=input[i].opd1;  result=input[i].result;  }    for(int j=i;j<no\_of\_lines;j++)  {  if((input[j].opd1==(result)))  {  input[j].opd1=temp;  }  else if(input[j].opd2==(result))  {  input[j].opd2==temp;  }    }  }    }  }  }  void dead\_code(int no\_of\_lines)  {  char temp;  int flag=0;  //Dead code elemination  for(int i=0;i<no\_of\_lines;i++)  {  temp=input[i].result;  //cout<<"Result in temp: "<<temp;  for(int j=1;j<no\_of\_lines;j++)  {  if(i==j)  {  continue;  }  else if((input[j].opd1)==temp||(input[j].opd2)==temp)  {  flag++;  //cout<<"Matching condition for: "<<temp<<endl;  }    }  if(flag==0)  {  //cout<<"In flag\n";  input[i].opr='x';  flag=0;  }    }  }  void display(int no\_of\_lines)  {  for(int i=0;i<no\_of\_lines;i++)  {  cout<<endl<<input[i].opr<<"\t"<<input[i].opd1<<"\t"<<input[i].opd2<<"\t"<<input[i].result;  }  }  void display\_after(int no\_of\_lines)  {  for(int i=0;i<no\_of\_lines;i++)  {  if(input[i].opr=='x')  {  continue;  }  else  {  cout<<endl<<input[i].opr<<"\t"<<input[i].opd1<<"\t"<<input[i].opd2<<"\t"<<input[i].result;  }  }  }  int main()  {  int n,no\_of\_lines,flag=0;  cout<<"Enter number of three address codes to take: ";  cin>>n;  for(int i=0;i<n;i++)  {  cout<<"Enter Operator: ";  cin>>input[i].opr;  cout<<"\nEnter first Operand: ";  cin>>input[i].opd1;  cout<<"\nEnter second Operand: ";  cin>>input[i].opd2;  cout<<"\nEnter Result: ";  cin>>input[i].result;  cout<<endl;    }  code\_optimization(n);    return 0;  } |