```
% HENG LOW WEE
% U096901R
% Tut 5 Ex 4

:- op(1099,yf,;).
:- op(960,fx,if).
:- op(959,xfx,then).
:- op(958,xfx,else).
:- op(960,fx,while).
:- op(959,xfx,do).
:- op(600,xfx,@).

compileExpr(K,E,E,T,T) :-
    integer(K),!,
    write('    esp -= 4 ; *(int*)&M[esp] = '),
    write(K),write(' ; // push '), writeln(K).
compileExpr(V,Ein,Eout,Tin,Tout) :-
    atom(V),!,
    (   member((V->Addr),Ein)
    ->  Tout = Tin, Eout = Ein
    ;   Tout is Tin+4, Eout = [(V->Tin)|Ein], Addr = Tin),
    write('    ecx = *(int*)&M['),
    write(Addr),
    write('] ; esp -= 4 ; *(int*)&M[esp] = ecx ; // push '),
    writeln(V).

% Arrays as expressions <- part of the solution to T5E4
compileExpr(A@I,Ein,Eout,Tin,Tout) :-
    atom(A), !,
    (   member((A->Addr),Ein)
    ->  Taux = Tin, Eaux = Ein
    ;   Taux is Tin+400, Eaux = [(A->Tin)|Ein], Addr = Tin),
    compileExpr(I,Eaux,Eout,Taux,Tout),
        writeln('    ebx = *(int*)&M[esp] ; esp += 4 ;'),
    writeln('    ebx *= 4;') ,
    write('    ebx += '),write(Addr),writeln(' ;'),
    writeln('    eax = *(int*)&M[ebx] ;'),
    writeln('    esp -= 4 ; *(int*)&M[esp] = eax ; ').
compileExpr(Exp,Ein,Eout,Tin,Tout) :-
    Exp =.. [O,A,B],
    compileExpr(A,Ein,Eaux,Tin,Taux),
    compileExpr(B,Eaux,Eout,Taux,Tout),
    writeln('    ecx = *(int*)&M[esp] ; esp += 4 ;'),
    writeln('    eax = *(int*)&M[esp] ; esp += 4 ;'),
    write('    eax '), write(O), writeln('= ecx ;'),
```

```
    write('    esp -= 4 ; *(int*)&M[esp] = eax ; // push result of '),
    writeln(O).

% Array assignments as statements <- part of the solution to T5E4
compile(A@I=E,Ein,Eout,Tin,Tout,L,L) :-
    atom(A),!,
    compileExpr(E,Ein,Ea1,Tin,Ta1),
    compileExpr(I,Ea1,Ea2,Ta1,Ta2),
    (   member((A->Addr),Ea2)
    ->  Tout = Ta2, Eout = Ea2
    ;   Tout is Ta2+400, Eout = [(A->Ta2)|Ea2], Addr = Ta2),
        writeln('    ebx = *(int*)&M[esp] ; esp += 4 ;'),
    writeln('    ebx *= 4;') ,
    write('    ebx += '),write(Addr),writeln(' ;'),
    writeln('    edx = *(int*)&M[esp] ; esp += 4 ;'),
    writeln('    *(int*)&M[ebx] = edx ; ' ) .
compile(V=E,Ein,Eout,Tin,Tout,L,L) :-
    compileExpr(E,Ein,Eaux,Tin,Taux),
    (   member((V->Addr),Eaux)
    ->  Tout = Taux, Eout = Eaux
    ;   Tout is Taux+4, Eout = [(V->Taux)|Eaux], Addr = Taux),
    writeln('    ecx = *(int*)&M[esp] ; esp += 4 ;'),
    write('    *(int*)&M['),write(Addr),write('] = ecx ; // pop '),
    writeln(V).
compile(if B then S1 else S2,Ein,Eout,Tin,Tout,Lin,Lout) :- !,
    B =.. [O,X,Y], La1 is Lin+1,
    (   O == (\=) -> Otrans = '!=' ; Otrans = O ),
    writeln('    // start of if-then-else statement'),
    compileExpr(X,Ein,Ea1,Tin,Ta1),
    compileExpr(Y,Ea1,Ea2,Ta1,Ta2),
    writeln('    ecx = *(int*)&M[esp] ; esp += 4 ;') ,
    writeln('    eax = *(int*)&M[esp] ; esp += 4 ;') ,
    write('    if ( eax '), write(Otrans),
    write(' ecx ) goto Lthen'), write(Lin), writeln('; // if condition'),
    compile(S2,Ea2,Ea3,Ta2,Ta3,La1,La2),
    write('    goto Lendif'),write(Lin),writeln(';'),
    write('Lthen'),write(Lin),writeln(':'),
    compile(S1,Ea3,Eout,Ta3,Tout,La2,Lout),
    write('Lendif'),write(Lin),writeln(':').
compile(if B then S,Ein,Eout,Tin,Tout,Lin,Lout) :- !,
    B =.. [O,X,Y], La1 is Lin+1,
    (   O == (\=) -> Otrans = '!=' ; Otrans = O ),
    writeln('    // start of if-then statement'),
    compileExpr(X,Ein,Ea1,Tin,Ta1),
    compileExpr(Y,Ea1,Ea2,Ta1,Ta2),
```

```
    writeln('    ecx = *(int*)&M[esp] ; esp += 4 ;') ,
    writeln('    eax = *(int*)&M[esp] ; esp += 4 ;') ,
    write('    if ( eax '), write(Otrans),
    write(' ecx ) goto Lthen'), write(Lin), writeln('; // if condition'),
    write('    goto Lendif'),write(Lin),writeln(';'),
    write('Lthen'),write(Lin),writeln(':'),
    compile(S,Ea2,Eout,Ta2,Tout,La1,Lout),
    write('Lendif'),write(Lin),writeln(':').
compile(while B do S,Ein,Eout,Tin,Tout,Lin,Lout) :- !,
    B =.. [O,X,Y], La1 is Lin+1,
    (   O == (\=) -> Otrans = '!=' ; Otrans = O ),
    write('Lwhile'),write(Lin),writeln(':'),
    compileExpr(X,Ein,Ea1,Tin,Ta1),
    compileExpr(Y,Ea1,Ea2,Ta1,Ta2),
    writeln('    ecx = *(int*)&M[esp] ; esp += 4 ;') ,
    writeln('    eax = *(int*)&M[esp] ; esp += 4 ;') ,
    write('    if ( eax '), write(Otrans),
    write(' ecx ) goto Lwhilebody'), write(Lin), writeln(';'),
    write('    goto Lendwhile'),write(Lin),writeln(';'),
    write('Lwhilebody'),write(Lin),writeln(':'),
    compile(S,Ea2,Eout,Ta2,Tout,La1,Lout),
    write('    goto Lwhile'),write(Lin),writeln(';'),
    write('Lendwhile'),write(Lin),writeln(':').
compile(S1;S2,Ein,Eout,Tin,Tout,Lin,Lout) :- !,
    compile(S1,Ein,Eaux,Tin,Taux,Lin,Laux),
    compile(S2,Eaux,Eout,Taux,Tout,Laux,Lout).
compile(S;,Ein,Eout,Tin,Tout,Lin,Lout) :- !,
    compile(S,Ein,Eout,Tin,Tout,Lin,Lout).
compile({S},Ein,Eout,Tin,Tout,Lin,Lout) :- !,
    compile(S,Ein,Eout,Tin,Tout,Lin,Lout).

compileProg(P) :-
    writeln('#include <stdio.h>'),
    writeln('int eax,ebx,ecx,edx,esi,edi,ebp,esp;'),
    writeln('unsigned char M[10000];'),
    writeln('void exec(void) {'),
    compile(P,[],Eout,0,_,0,_),
    writeln('{}}'),nl,
    writeln('int main() {'),
    writeln('    esp = 10000 ;'),
    writeln('    exec();'),
    outputVars(Eout),
    writeln('    return 0;'),
    writeln('}').
```

```
outputVars([]).
outputVars([(V->Addr)|T]) :-
    write('    printf("'),write(V),write('=%d\\n",'),
    write('*(int*)&M['),write(Addr),writeln(']);'),
    outputVars(T).


:- P = ( a@0 = 4; a@1 = 3 ; a@2=10; a@3 = 5 ; a@4 = -1 ;
     min = 10000 ;
         i = 0 ;
         while i < 5 do {
        if ( min > a@i ) then { min = a@i } ;
        i = i+ 1
            }
       ), compileProg(P).
```