```
// HENG LOW WEE
// U096901R
// Tut 7 Ex 1

#include <stdio.h>

int eax, ebx, ecx, edx, esi, edi, ebp, esp;
unsigned char M[10000];

void exec() {
esp = 10000;

esp -= 4 ; *(int*)&M[esp] = eax ;
esp -= 4 ; *(int*)&M[esp] = ecx ;
esp -= 4 ; *(int*)&M[esp] = edx ;
esp -= 4 ; *(int*)&M[esp] = 5 ;
esp -= 4 ; *(int*)&M[esp] = 10 ;
eax = (int) && return_address ;
esp -= 4 ; *(int*)&M[esp] = eax ;
goto f ;

f:
    esp -= 4 ; *(int*)&M[esp] = ebp ; // push ebp
    ebp = esp ;
    esp -= 4 ; *(int*)&M[esp] = ebx ; // push ebx
    esp -= 4 ; *(int*)&M[esp] = edi ; // push edi
    esp -= 4 ; *(int*)&M[esp] = esi ; // push esi
    esp -= 8 ; // allocate space for local vars

    // get value of n and k
    ebx = *(int*)&M[ebp+8] ;
    edi = *(int*)&M[ebp+12] ;
    if (ebx == 0) goto then_branch;
    if (ebx == edi) goto then_branch;
    goto skip;

    then_branch:
        eax = 1;
        goto exit_f;

skip:
    esp -= 4 ; *(int*)&M[esp] = eax ; // push eax
    esp -= 4 ; *(int*)&M[esp] = ecx ; // push ecx
    esp -= 4 ; *(int*)&M[esp] = edx ; // push edx
    eax = *(int*)&M[ebp+12] ;
```

```
    esp -= 4 ; *(int*)&M[esp] = eax ; // push k
    eax = *(int*)&M[ebp+8] ;
    eax -= 1 ;
    esp -= 4 ; *(int*)&M[esp] = eax ; // push n-1
    eax = (int) && return_address1 ;
    esp -= 4 ; *(int*)&M[esp] = eax ; // push return addr
    goto f;
return_address1:
    esp += 8 ; // clear arguments
    edx = *(int*)&M[esp] ; esp += 4 ; // restore edx
    ecx = *(int*)&M[esp] ; esp += 4 ; // restore ecx
    *(int*)&M[ebp-16] = eax ; // x = return value
    eax = *(int*)&M[esp] ; esp += 4 ; // restore eax

    esp -= 4 ; *(int*)&M[esp] = eax ; // push eax
    esp -= 4 ; *(int*)&M[esp] = ecx ; // push ecx
    esp -= 4 ; *(int*)&M[esp] = edx ; // push edx
    eax = *(int*)&M[ebp+12] ;
    eax -= 1 ;
    esp -= 4 ; *(int*)&M[esp] = eax ; // push k-1
    eax = *(int*)&M[ebp+8] ;
    eax -= 1 ;
    esp -= 4 ; *(int*)&M[esp] = eax ; // push n-1
    eax = (int) && return_address2 ;
    esp -= 4 ; *(int*)&M[esp] = eax ; // push return addr
    goto f;
return_address2:
    esp += 8 ; // clear arguments
    edx = *(int*)&M[esp] ; esp += 4 ; // restore edx
    ecx = *(int*)&M[esp] ; esp += 4 ; // restore edx
    *(int*)&M[ebp-20] = eax ; // y = return value
    eax = *(int*)&M[esp] ; esp += 4 ; // restore eax
    eax = *(int*)&M[ebp-16] ; // load LHS
    eax += *(int*)&M[ebp-20] ; // add RHS, return value set now

exit_f:
    esp += 8 ; // clear local vars
    esi = *(int*)&M[esp] ; esp += 4 ; // restore
    edi = *(int*)&M[esp] ; esp += 4 ; // callee
    ebx = *(int*)&M[esp] ; esp += 4 ; // registers
    ebp = *(int*)&M[esp]; esp += 4 ;
    esp += 4 ; goto * *(void**)&M[esp-4] ; // return

return_address: {}
}
```

```
int pascal (int n, int k) {
    if (n==0 || n==k) return 1;
    return pascal(n-1,k) + pascal(n-1,k-1);
}

int main() {
    printf("C pascal(10,5) = %d\n", pascal(10,5));
    exec();
    printf("eax = %d\n", eax);
    return 0;
}
```