دانشکده‌ی مهندسی کامپیوتر

طراحی کامپایلرها

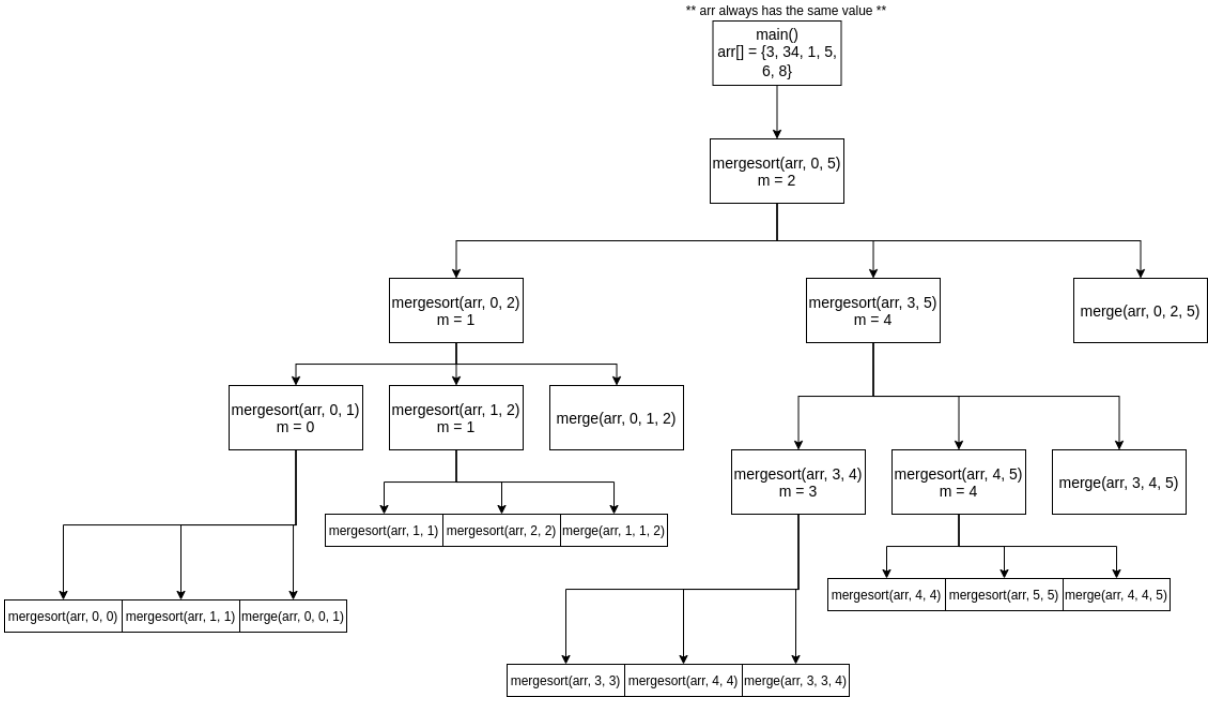
نیم‌سال دوم ۹۹ـ۹۸

نام و نام خانوادگی: حسن ذاکر، علیرضا دقیق، سپهر فعلی

پاسخ تمرین سری پنجم — بهینه‌سازی و تولید کد — موعد تحویل: ۹۹/۱۰/۲۷

مسئله‌ی ۱.

پاسخ.

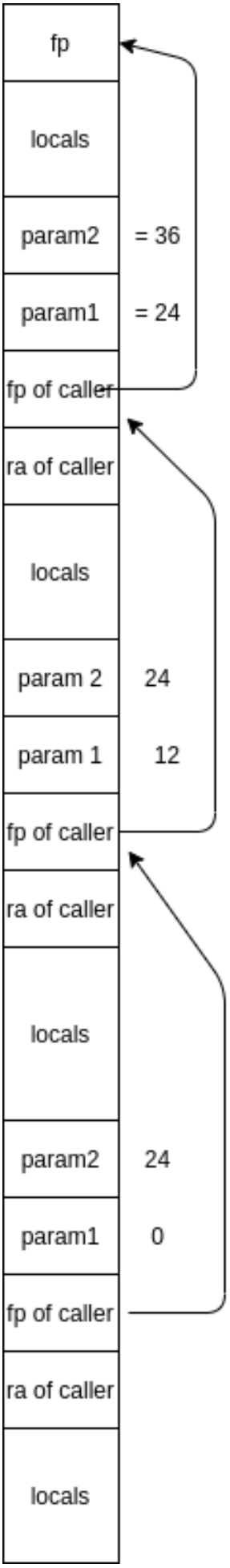


مسئله‌ی ۲.

پاسخ. ابتدا می‌آییم کد tac مربوط به آنرا مینویسیم:

```
1  _main;
2    BeginFunc 20
3    t0 = 24
4    a = t0
5    t1 = 36
6    b = t1
7    pushParams t1
8    pushParams t0
9    t2 = Lcall _gcc
10   popParams 8
11   endfunc
```

```
12 _gcc
13  beginfunc 16
14  t0 = x
15  t1 = y
16  ifz t0 goto L0
17  t2 = y \% x
18  r = t2
19  pushParams t0
20  pushparams t2
21  lcall gcc
22  popParams 8
23  endfunc
24  L0:
25   return t1
```


fp
locals
param2
= 36
param1
= 24
fp of caller
ra of caller
locals
param 2
24
param 1
12
fp of caller
ra of caller
locals
param2
24
param1
0
fp of caller
ra of caller
locals

اولین بار که خط یازده اجرا میشود با توجه به استک فریم در ۱۶ * ۴ تا پایین تر کال میشود و مسوول کردنpush آن پارامتر ها تابع ایgcd است که اولین بار کال میشود. اولین بار که خط هشت کال میشود fp در ۲۵ * ۴ تا پایین تر است و مسوول ؛صدا کردن این تابع gcd ای است که قبل از آن کال شده است و مقدار را به تابع بالاتر میدهد و پاپ پارامز را انجام میدهد.

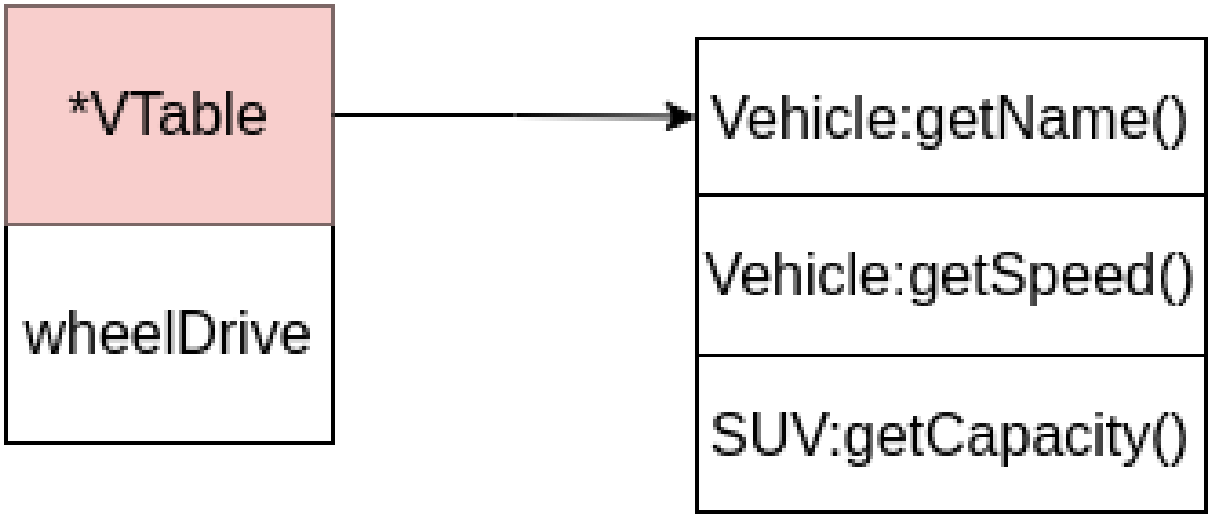
مسئله‌ی ۳.

پاسخ.

- call by value
  1, 1, 0, 0, 1, 2, 3, 4
  1, 0, 1, 2, 3, 4
- call by reference
  1, 1, 0, 0, 1, 2, 3, 4
  1, 0, 1, 2, 3, 4
- call by name
  0, 0, 0, 0, 1, 2, 3, 4
  1, 0, 1, 2, 3, 4
- call by value-result
  1, 1, 0, 0, 1, 2, 3, 4
  1, 0, 1, 2, 3, 4

مسئله‌ی ۴.

پاسخ.



-

مسئله‌ی ۵.

پاسخ.

```
_A.methodA:
    BeginFunc 16;
    _t0 = *(this + 4);
    _t1 = a + _t0;
    _t2 = 10;
    _t3 = _t1 * _t2;
    _v0 = _t3;  //register that keeps return value
    EndFunc;
VTable A = _A.methodA, ;


_B.methodB :
    BeginFunc 28;
    _t0 = *(this + 8);
    _t1 = param + _t0;
    _t2 = *(this);
    _t3 = *(_t2);
    _t4 = 6;
    PushParam _t4;
    _t5 = ACall _t3;
    PopParams 4;
    _t6 = _t1 * _t5;
    _v0 = _t6;
    EndFunc;
VTable B = _A.methodA, _B.methodB, ;

main:
    BeginFunc 28;
    _t0 = 12;
    PushParam _t0;
    b = LCall_Alloc;
    PopParams 4;
    _t1 = B();
    *b = _t1;
    _t0 = 5;
    *(b + 4) = _t0;
    _t0 = 10;
    *(b + 8) = _t0;
    _t0 = 4;
    PushParam _t0;
    x = LCall_Alloc;
    PopParams 4;
    _t1 = *(b)  // pointer to vtable
    _t2 = *(_t1 + 4) //pointer to _B.methodB in vtable B
    _t0 = 2;
    PushParam _t0;
    *x = ACall _t2;
    PopParams 4;
```

```
49          EndFunc;
```

مسئله‌ی ۶.

پاسخ.

Instruction	Live Variables
a = 1 + 2;	b, e, f
b = a + b;	a, b, e, f
z = a * 2;	a, b, e, f
c = b + e;	a, b, e, f
d = c + b;	a, b, c, f
x = b + 3;	a, b, c, d, f
z = a * 8;	a, c, d, f, x
t = c - 2;	c, d, f, x, z
f = x + f;	d, f, x, z
y = x - 2;	d, x, z
d = d - y;	d, x, y, z
	d, x, z

۱.

۲. در دور اول، دستورات زیر به ترتیب حذف میگردند. (با توجه به آنالیز متغیرهای زنده بخش قبلی)

(آ) ابتدا دستور f = x + f; حذف میگردد. زیرا بعد از آن، متغیر f زنده نخواهدبود.

(ب) سپس دستور t = c - 2; حذف میگردد. زیرا بعد از آن، متغیر t زنده نخواهدبود.

(ج) سپس دستور z = a * 2; حذف میگردد. زیرا بعد از آن، متغیر z زنده نخواهدبود.

نتیجه این تغییرات و آنالیز دوباره متغیرهای زنده به شکل زیر خواهد بود:

Instruction	Live Variables
a = 1 + 2;	b, e
b = a + b;	a, b, e
c = b + e;	a, b, e
d = c + b;	a, b, c
x = b + 3;	a, b, d
z = a * 8;	a, d, x
y = x - 2;	d, x, z
d = d - y;	d, x, z
	d, x, z

در دور دوم، هیچ دستوری با لحاظ کردن متغیرهای زنده حذف نمی‌شود. پس به سراغ مقادیر ثابت میرویم و Copy Propagation

(آ) ابتدا a = 1 + 2; تبدیل میشود به a = 3;

(ب) سپس b = a + b; تبدیل میشود به b = 3 + b;

(ج) سپس z = a * 8; تبدیل میشود به z = 24;

نتیجه این تغییرات و آنالیز دوباره متغیرهای زنده به شکل زیر خواهد بود:

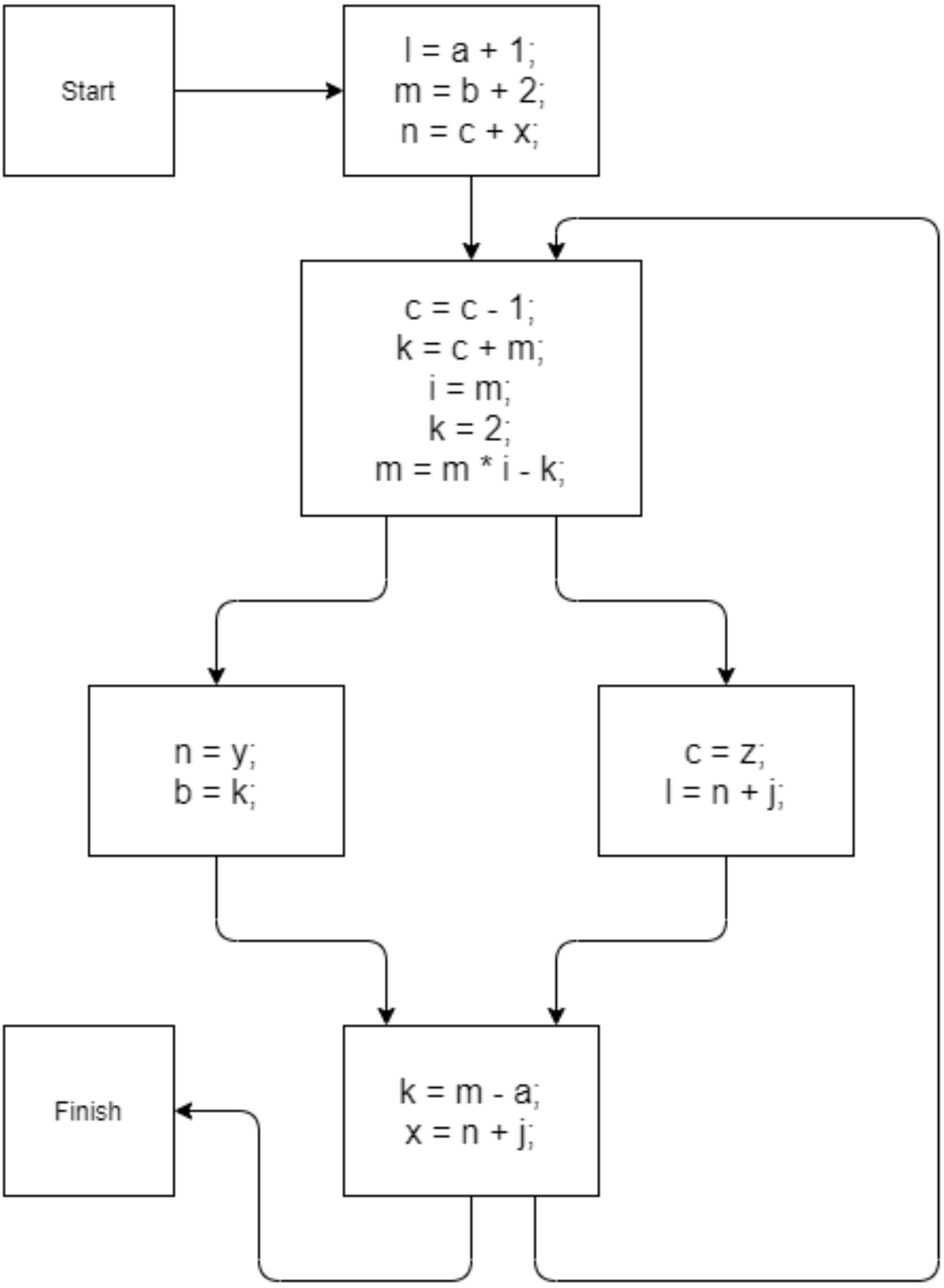
Instruction	Live Variables
a = 3;	b, e
b = 3 + b;	b, e
c = b + e;	b, e
d = c + b;	b, c
x = b + 3;	b, d
z = 24;	d, x
y = x - 2;	d, x, z
d = d - y;	d, x, z
	d, x, z

در دور سوم و آخر، با توجه به متغیرهای زنده، فقط میتوان اولین جمله را حذف کرد. یعنی a = 3; کد نهایی:

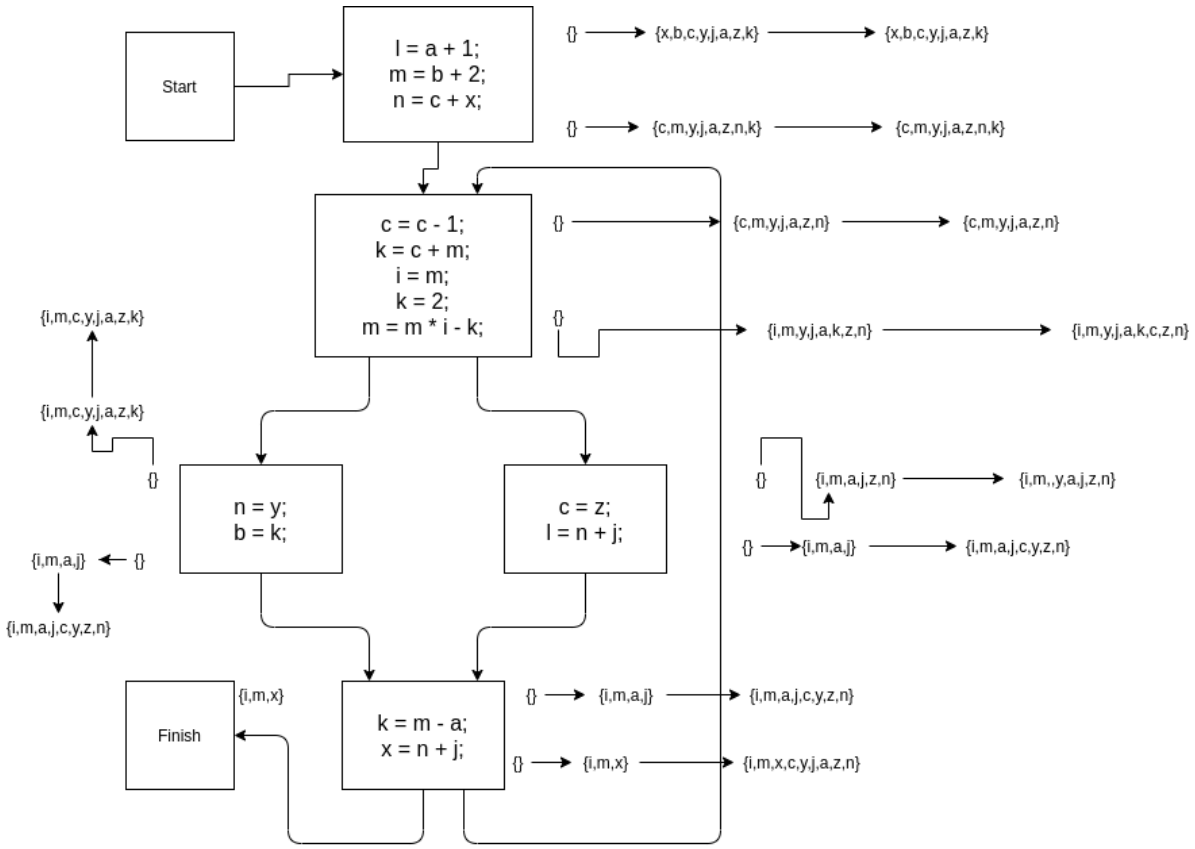
```
1 b = 3 + b;
2 c = b + e;
3 d = c + b;
4 x = b + 3;
5 z = 24;
6 y = x - 2;
7 d = d - y;
```

مسئله‌ی ۷.

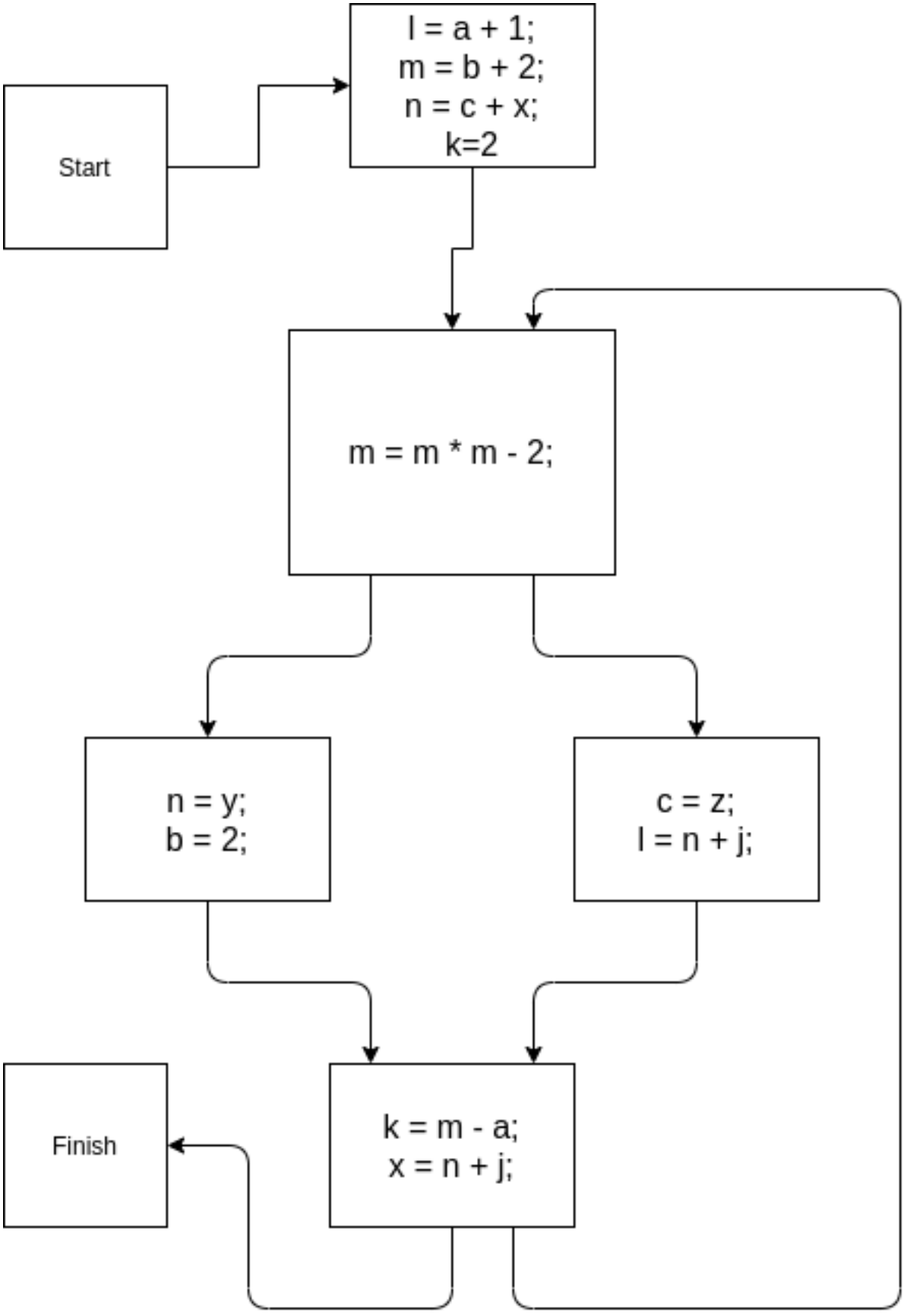
پاسخ.



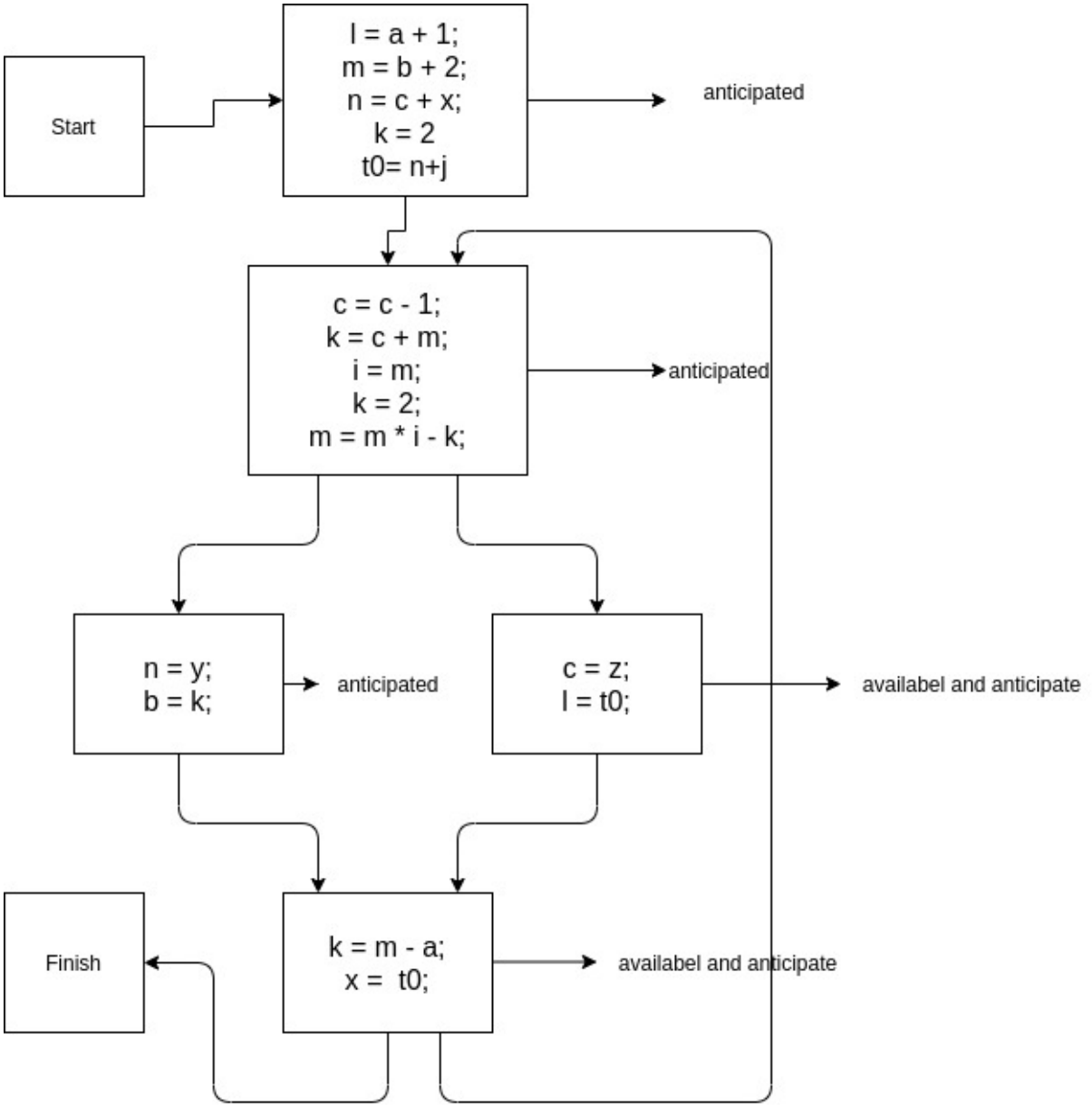
.١


Start
l = a + 1;
m = b + 2;
n = c + x;
{}
{x,b,c,y,j,a,z,k}
{x,b,c,y,j,a,z,k}
{}
{c,m,y,j,a,z,n,k}
{c,m,y,j,a,z,n,k}
c = c - 1;
k = c + m;
i = m;
k = 2;
m = m * i - k;
{}
{c,m,y,j,a,z,n}
{c,m,y,j,a,z,n}
{}
{i,m,y,j,a,k,z,n}
{i,m,y,j,a,k,c,z,n}
{i,m,c,y,j,a,z,k}
{i,m,c,y,j,a,z,k}
{}
n = y;
b = k;
c = z;
l = n + j;
{}
{i,m,a,j,z,n}
{i,m,,y,a,j,z,n}
{}
{i,m,a,j}
{i,m,a,j,c,y,z,n}
{i,m,a,j}
{}
{i,m,a,j,c,y,z,n}
Finish
{i,m,x}
k = m - a;
x = n + j;
{}
{i,m,a,j}
{i,m,a,j,c,y,z,n}
{}
{i,m,x}
{i,m,x,c,y,j,a,z,n}

.٢



.۳



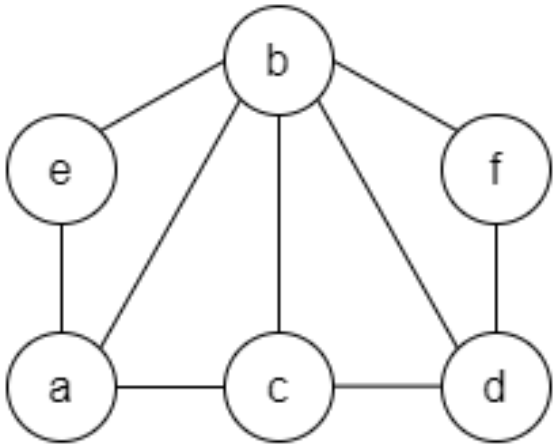
.۴

مسئله‌ی ۸.

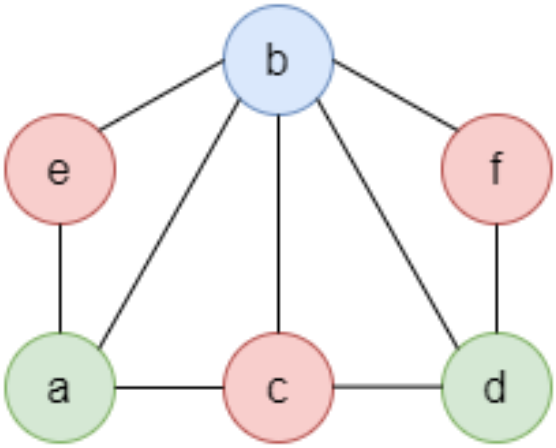
پاسخ.

Instruction	Live Variables
a = b	b, e
c = 7 + 7 * e	a, b, e
d = a	a, b, c
a = d * d	b, c, d
d = 5 * a	a, b, c
f = c * 5 + 10	b, c, d
f = d - f	b, d, f
c = f + 1	b, f
e = c * b	b, c
	b, c

۱. باتوجه به جدول بالا، گراف تداخل رجیسترها به شکل زیر خواهد بود:



۲. با توجه به این که در گراف، دور به طول ۳ داریم، در بهترین حالت سه رجیستر نیاز است. سه عدد رجیستر را امتحان میکنیم تا ببینیم کفایت می‌کند یا خیر. که خواهیم دید کفایت می‌کند.
ابتدا e را برمیداریم. سپس a را برمیداریم. بعد c و بعد d را برمیداریم. تا اینجا، هر راسی را که برداشتیم، دو عدد یال خروجی داشت. حال f و در آخر b را برمیداریم.
حال شروع به رنگ کردن می‌کنیم. ابتدا b را میاوریم و آبی میکنیم. سپس f را قرمز میکنیم و بعد آن، d را سبز. بعدی نوبت c است. آن را قرمز میکنیم. بعدی a است و سبز میشود. و آخری که e خواهدبود، قرمز میگردد.
تصویر پایین، نتیجه پایانی را نشان میدهد:



مسئله‌ی ۹.

پاسخ.

I	H	G	F	E	D	C	B	A	variable
۱	۲	۱	۱	۱	۰	۱	۰	۱	refcount

-

I	H	G	F	E	D	C	B	A	variable
۱	۲	۱	۱	۱	۰	۱	۰	۱	refcount

-

- به روش stop and copy عمل می‌کنیم و چون هیچ از A به سایر نودها وجود نخواهد داشت، نودهای باقی‌مانده CEFGHI (که به هم ارجاع دارند و یک دورمانند ایجاد می‌کنند) در این روش حذف می‌شوند و خانه‌ی A به سمت old space منتقل می‌شود پس مقدار آدرس A به اندازه طول هر سمت کم یا زیاد می‌شود(بسته با این که old space در سمت چپ یا راست باشد).