



دانشگاه صنعتی امیرکبیر  
دانشکده مهندسی کامپیوتر

گزارش تکلیف پنجم درس پردازش تصویر رقمی

دانشجو:

سید احمد نقوی نوزاد

ش-د: ۹۴۱۳۱۰۶۰

استاد:

دکتر رحمتی

بهار ۹۵

## LZW Dictionary Coder

قسمت الف)

با توجه به خروجی رمزگذار LZW و دیکشنری اولیه و نیز الگوریتم رمزگشای LZW که در ادامه می‌آید داریم:

### Initial Dictionary

Index	Entry
1	a
2	-
3	r
4	t

### Output of LZW Encoder (To be Decoded!)

3	1	4	6	8	4	2	1	2	5	10	6	11	13	6
---	---	---	---	---	---	---	---	---	---	----	---	----	----	---

## LZW Decoding Algorithm

1. Read OLD\_CODE ('CODE' equals 'Dictionary Index')
2. output translation of OLD\_CODE
3. CHARACTER = translation of OLD\_CODE
4. WHILE there are still input characters DO
5.     Read NEW\_CODE
6.     IF NEW\_CODE is not in the translation table THEN
7.         STRING = get translation of OLD\_CODE
8.         STRING = STRING+CHARACTER
9.     ELSE
10.         STRING = get translation of NEW\_CODE
11.     END of IF
12.     output STRING
13.     CHARACTER = first character in STRING
14.     add translation of OLD\_CODE + CHARACTER to the translation table
15.     OLD\_CODE = NEW\_CODE
16. END of WHILE

Input Indices	3	1	4	6	8	4	2	1	2	5	10	6	11	13	6
Translation Of index (Decoder Output)	r	a	t	at	ata	t	-	a	-	ra	t-	at	-a	-r	at
String added to dictionary	<null>	ra	at	ta	ata	atat	t-	-a	a-	-r	rat	t-a	at-	-a-	-ra

### Final Dictionary After Decoding Procedure

Index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Entry	a	-	r	t	ra	at	ta	ata	atat	t-	-a	a-	-r	rat	t-a	at-	-a-	-ra

## قسمت ب)

تمامی مراحل رمزگذاری رشته‌ی خروجی مرحله‌ی قبل در ادامه قابل مشاهده است:

### Input String

String	r	a	t	a	t	a	t	a	t	-	a	-	r	a	t	-	a	t	-	a	-	r	a	t
Pointer	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24

Pointer	1	2	3	4	6	9	10	11	12	13	15	17	19	21	23
Sequence read	r	a	t	at	ata	t	-	a	-	ra	t-	at	-a	-r	at
Dictionary Index (Encoder Output)	3	1	4	6	8	4	2	1	2	5	10	6	11	13	6
String added to dictionary	ra	at	ta	ata	atat	t-	-a	a-	-r	rat	t-a	at-	-a-	-ra	<null>

### Final Dictionary After Encoding Procedure

Index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Entry	a	-	r	t	ra	at	ta	ata	atat	t-	-a	a-	-r	rat	t-a	at-	-a-	-ra

همانطور که قابل مشاهده است، رشته‌ی خروجی کدگذار که شامل اندیس‌های دیکشنری می‌باشد، کاملاً با رشته‌ی ورودی قسمت الف یکسان است که این خود گواهی بر صحت پاسخ‌ها می‌باشد.

## جواب سوال ۲

### Huffman Coder

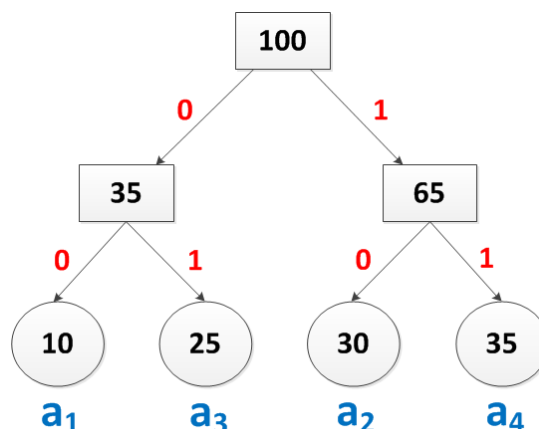
## قسمت الف)

$$A = \{a_1, a_2, a_3, a_4\}; \quad P(a_1)=.1, \quad P(a_2)=.3, \quad P(a_3)=.25, \quad P(a_4)=.35$$

$$H(X) = -\sum_i p_i \log_2 p_i = -(.1 \times \log_2 .1 + .3 \times \log_2 .3 + .25 \times \log_2 .25 + .35 \times \log_2 .35) = 1.8834$$

## قسمت ب)

Symbol	Codeword
a <sub>1</sub>	00
a <sub>2</sub>	10
a <sub>3</sub>	01
a <sub>4</sub>	11



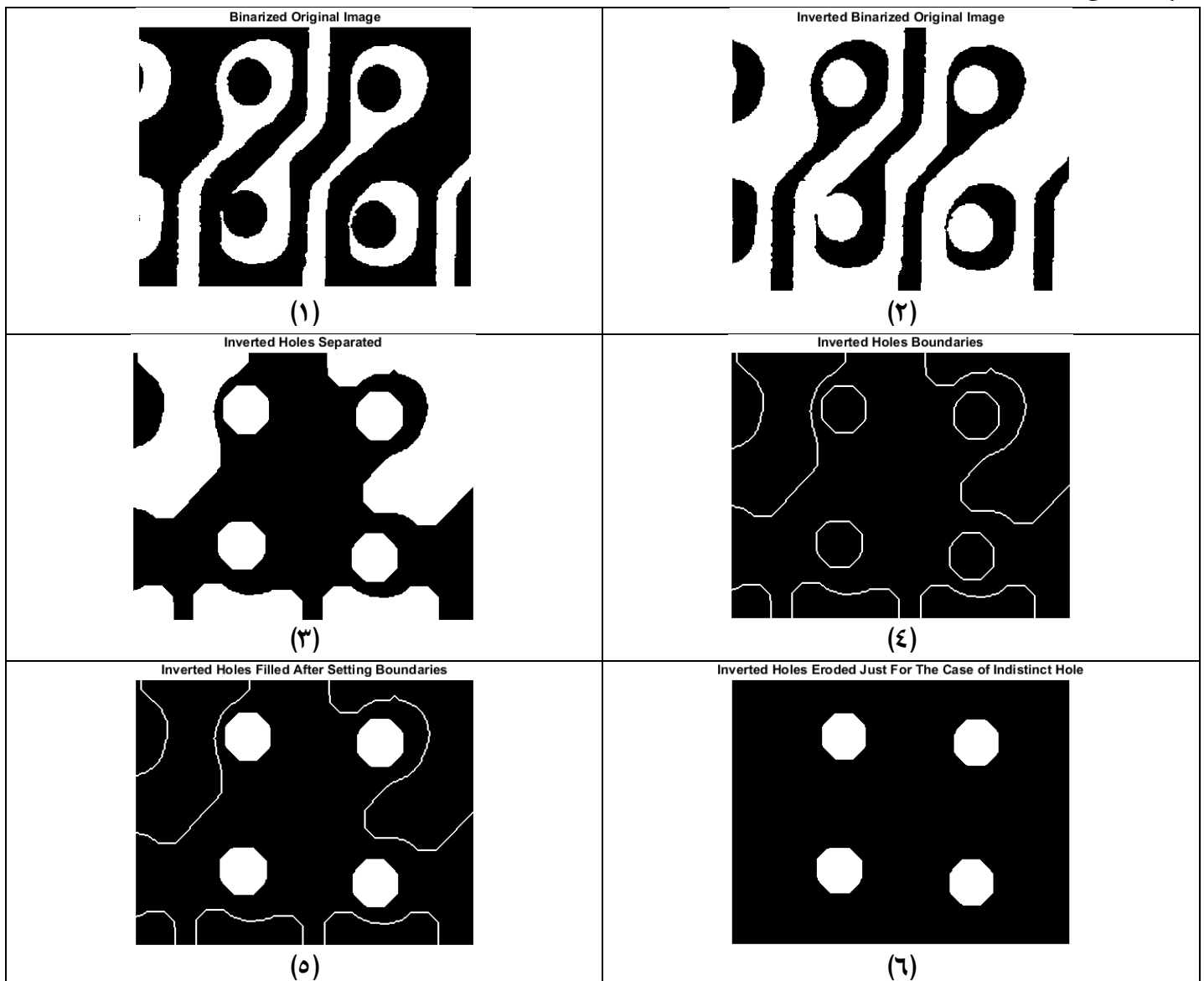
$$\bar{l} = \sum_i p_i l_i = 2 \times \sum_i p_i = 2 \times 1 = 2 \quad (\text{average code length})$$

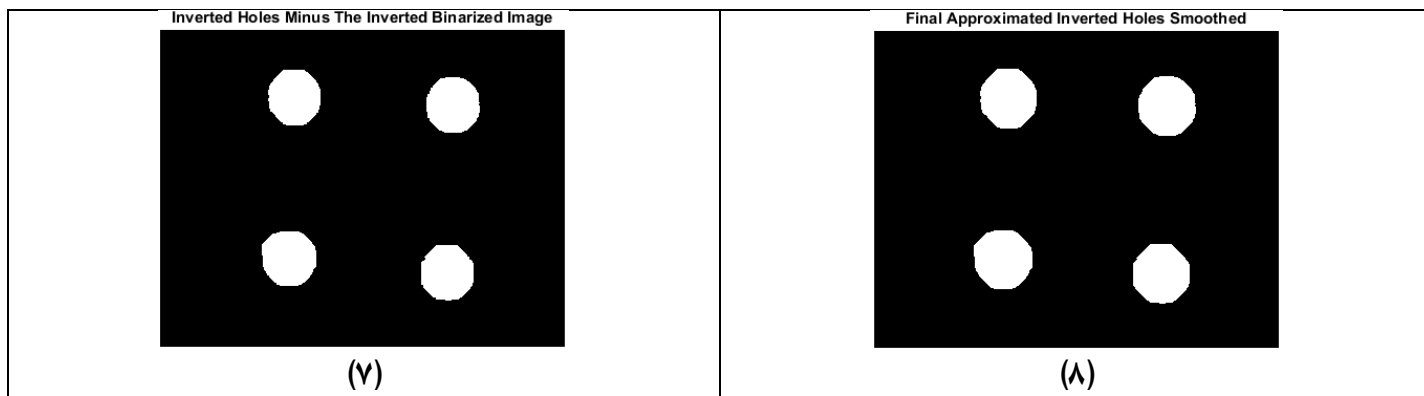
$$r = \bar{l} - H(X) = 2 - 1.8834 = .1166 \quad (\text{code redundancy})$$

## جواب سوال ۳

### Holes! Number and Diameter

در این قسمت جهت شمارش تعداد حفره‌ها و اندازه‌ی قطر آن‌ها متوسل به استفاده‌ی پیاپی از عملگرهای مورفولوژی خواهیم شد. در ابتدا با تعیین یک مقدار حدآستانه‌ی مناسب (با استفاده از تابع آماده‌ی `graythresh()`) تصویر `pcb.jpg` را تبدیل به یک تصویر باینری می‌نمائیم تا در ادامه بتوانیم از عملگرهای دودوئی مورفولوژی استفاده نمائیم. سپس چون قصد داریم تا حفره‌های موجود در تصویر را کشف نمائیم، تصویر دودوئی حاصله را معکوس می‌نمائیم تا در آن حفره‌ها با رنگ روشن (مقدار باینری برابر ۱) نشان داده شوند، و در ادامه چون یکی از حفره‌ها (حفره‌ی پایین سمت چپ) به طور کامل تفکیک‌شده نیست، لذا در تلاشیم تا با استفاده‌ی متوالی از عملگرهای `dilation` و `erosion` و نیز تکنیک‌هایی نظیر `boundary extraction`، هر چهار حفره را کاملاً تفکیک نمائیم که در نهایت به تصویری از حفره‌های تفکیک‌شده اما با اندازه‌های تقریبی خواهیم رسید. روند کار به صورت مرحله به مرحله در ادامه می‌آید:





نتایج نهائی در مورد اندازه‌ی حفره‌ها و مراکز آن‌ها به صورت زیر خواهد بود:

**Number of holes equals: 4**

Diameter of the hole with centroid coordinates [123.40, 218.06] is: **28.45**

Diameter of the hole with centroid coordinates [128.71, 64.86] is: **28.04**

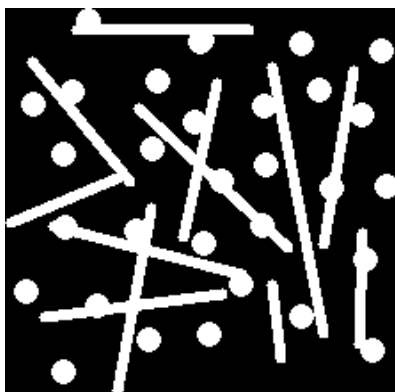
Diameter of the hole with centroid coordinates [274.36, 231.94] is: **28.09**

Diameter of the hole with centroid coordinates [279.92, 72.08] is: **28.33**

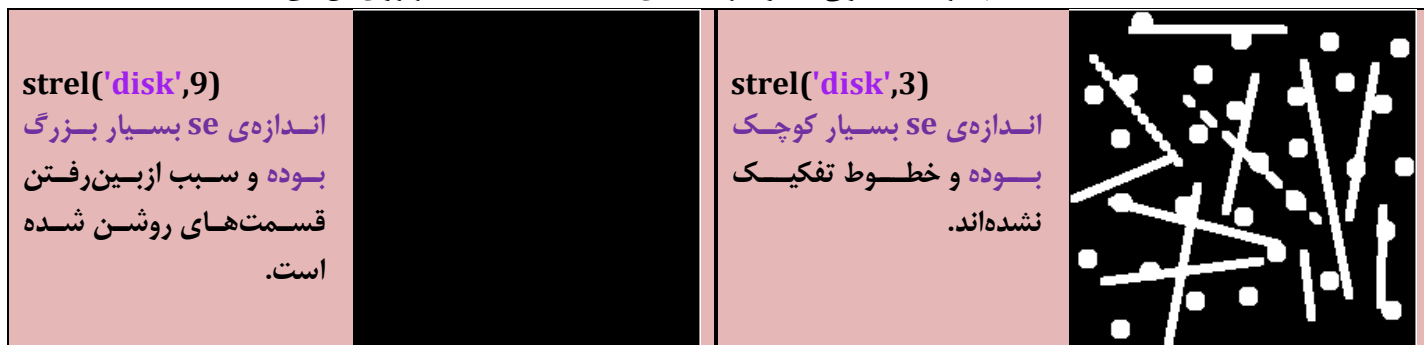
## جواب سوال ۴

### Separating out Lines and Holes

قسمت الف)



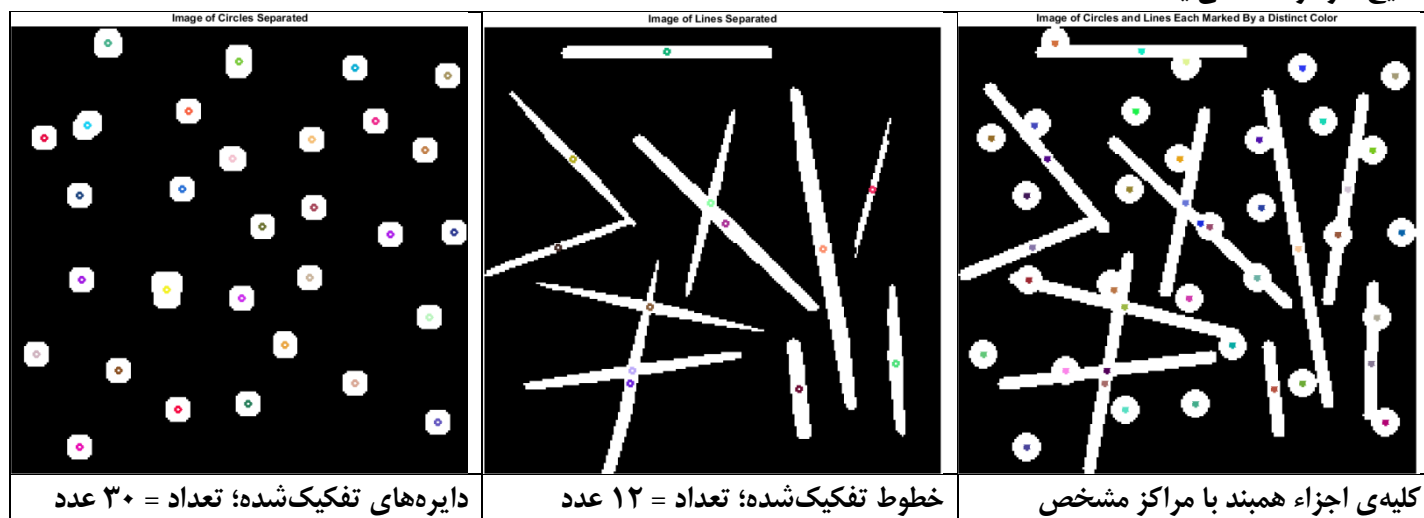
در تصویر بالا جهت تفکیک کردن دایره‌ها از خطوط، می‌بایست از عملگر مورفولوژی opening و البته پارامترهای مناسب برای انواع structural element استفاده نمائیم. در ادامه انواع se و نیز نتیجه‌ی اعمال opening بر روی آن می‌آید:



<p><code>strel('disk',6)</code>  اندازه‌ی <b>se</b> نسبتاً مناسب  بوده و سبب از بین رفتن کامل  خطوط شده و دایره‌ها نیز  تقریباً همان اندازه‌ی اصلی  خود را دارند.</p>		<p><code>strel('line',60,0)</code>  <b>se</b> از نوع خط مورب با  زاویه‌ی ۰ درجه و طول ۶۰  پیکسل، که تنها خطوط افقی  را جدا می‌کند.</p>	
<p><code>strel('line',60,10)</code>  <b>se</b> از نوع خط مورب با  زاویه‌ی ۱۰ درجه و طول ۶۰  پیکسل، که تنها خطوط با  زاویه‌ی تقریبی ۱۰ درجه را  جدا می‌کند.</p>		<p><code>strel('line',60,20)</code>  <b>se</b> از نوع خط مورب با  زاویه‌ی ۲۰ درجه و طول ۶۰  پیکسل، که تنها خطوط با  زاویه‌ی تقریبی ۲۰ درجه را  جدا می‌کند.</p>	
<p><code>strel('line',60,75)</code>  <b>se</b> از نوع خط مورب با  زاویه‌ی ۷۵ درجه و طول ۶۰  پیکسل، که تنها خطوط با  زاویه‌ی تقریبی ۷۵ درجه را  جدا می‌کند.</p>		<p><code>strel('line',40,95)</code>  <b>se</b> از نوع خط مورب با  زاویه‌ی ۹۵ درجه و طول ۴۰  پیکسل، که تنها خطوط با  زاویه‌ی تقریبی ۹۵ درجه را  جدا می‌کند.</p>	
<p><code>strel('line',60,135)</code>  <b>se</b> از نوع خط مورب با  زاویه‌ی ۱۳۵ درجه و طول  ۶۰ پیکسل، که تنها خطوط  با زاویه‌ی تقریبی ۱۳۵ درجه  را جدا می‌کند.</p>		<p><code>strel('line',60,170)</code>  <b>se</b> از نوع خط مورب با  زاویه‌ی ۱۷۰ درجه و طول  ۶۰ پیکسل، که تنها خطوط  با زاویه‌ی تقریبی ۱۷۰ درجه  را جدا می‌کند.</p>	
<p>تصویر نهائی از همگی  خطوط، که حاصل از اعمال  عملگر منطقی OR بر روی  همگی تصاویر خطوط حاصله  می‌باشد. اگرچه خطوط  نهائی کاملاً با خطوط موجود  در تصویر اصلی یکسان  نیستند، ولی حداقل از  دایره‌ها تفکیک شده‌اند!</p>		<p>در نهایت در مورد محدودیت‌های این روش می‌توان  گفت که برای جداسازی هر نوع شکل خاصی  می‌بایست از یک <b>se</b> مخصوص و البته تنظیم  پارامترهای مناسب بهره برد و مشابه این مسئله  نمی‌توان در آن واحد همگی اشکال را از یکدیگر  تفکیک نمود.</p>	

## قسمت ب)

تمامی مراحل مانند قسمت قبل می‌باشد، با این تفاوت که در این قسمت برای شمارش اجزاء همبند، از تابع درونی متلب با نام `bwlabel()` استفاده می‌نمائیم و البته برای حصول مراکز هر کدام از این نواحی از تابعی با نام `regionprops()` بهره می‌گیریم. در نهایت پس از رسم نواحی همبند برای هر کدام از اشکال دایره یا خط، مراکز آن‌ها را نیز با یک رنگ مجزا نمایش خواهیم داد. نتایج کار در ادامه می‌آید:



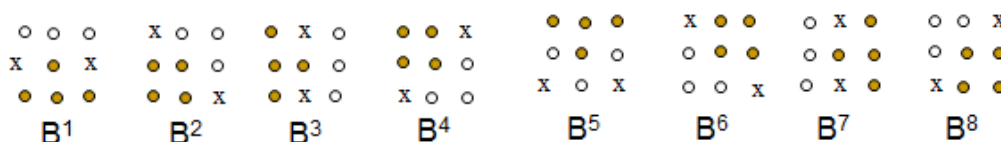
## جواب سوال ۵

### Skeleton Computing

## قسمت الف)

در ابتدا تصاویر ورودی را مانند سوالات قبلی، با انتخاب یک حد‌آستانه‌ی مناسب به تصاویر دودویی تبدیل می‌نمائیم و البته از معکوس‌شده‌ی آن‌ها استفاده می‌نمائیم، چرا که قسمت‌های مطلوب ما با رنگ تیره و مقدار منطقی ۰ در تصاویر اصلی نشان داده شده‌اند. برای محاسبه‌ی اسکلت تصاویر ورودی از الگوریتم **Thining** و **se** های مناسب مطابق زیر استفاده می‌نمائیم (لازم به ذکر است که در **se** ها، نقاط رنگی با ۱ و نقاط سفید با مقدار ۰ و **x** ها نیز با ۰ مقداردهی شده و در نهایت از تابع `bwhitmiss()` جهت استفاده در الگوریتم بهره‌برداری شده است):

- Basic idea:
- Use Hit-or-Miss operator as a sifter
  - Use multiple masks to characterize different patterns



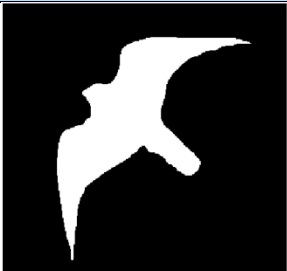
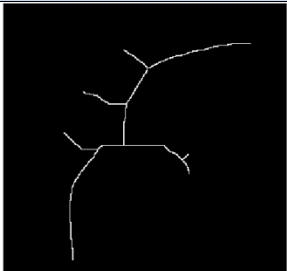

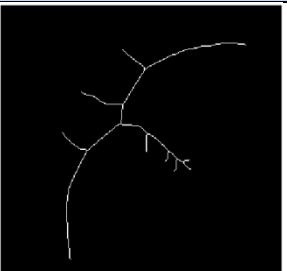

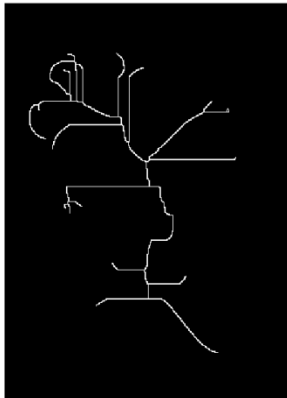

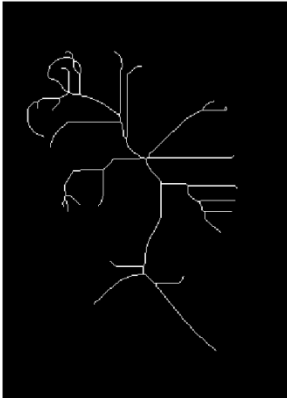
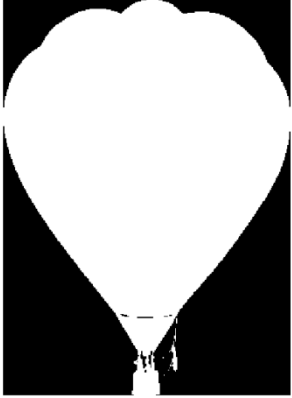
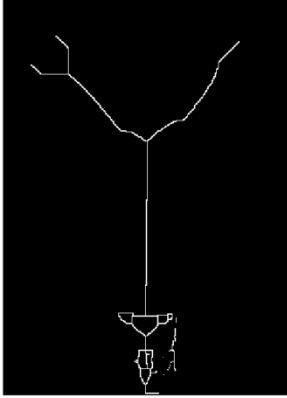
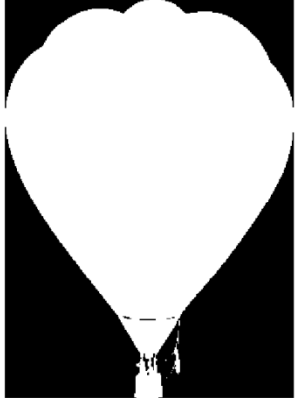
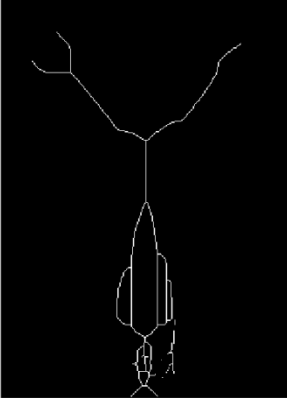

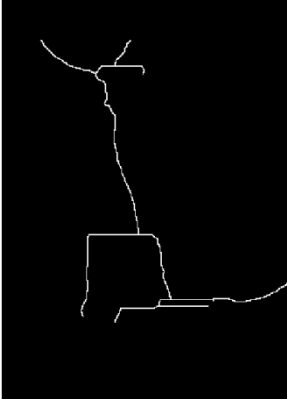

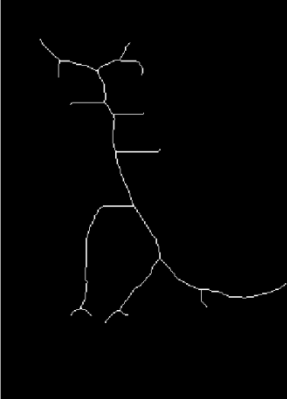
$$X_0 = X$$

$$X_k = (\dots ((X_{k-1} \otimes B^1) \otimes B^2 \dots \otimes B^8)$$

where  $X \otimes B = X - X \circledast B$

Stop the iteration when  $X_k = X_{k-1}$

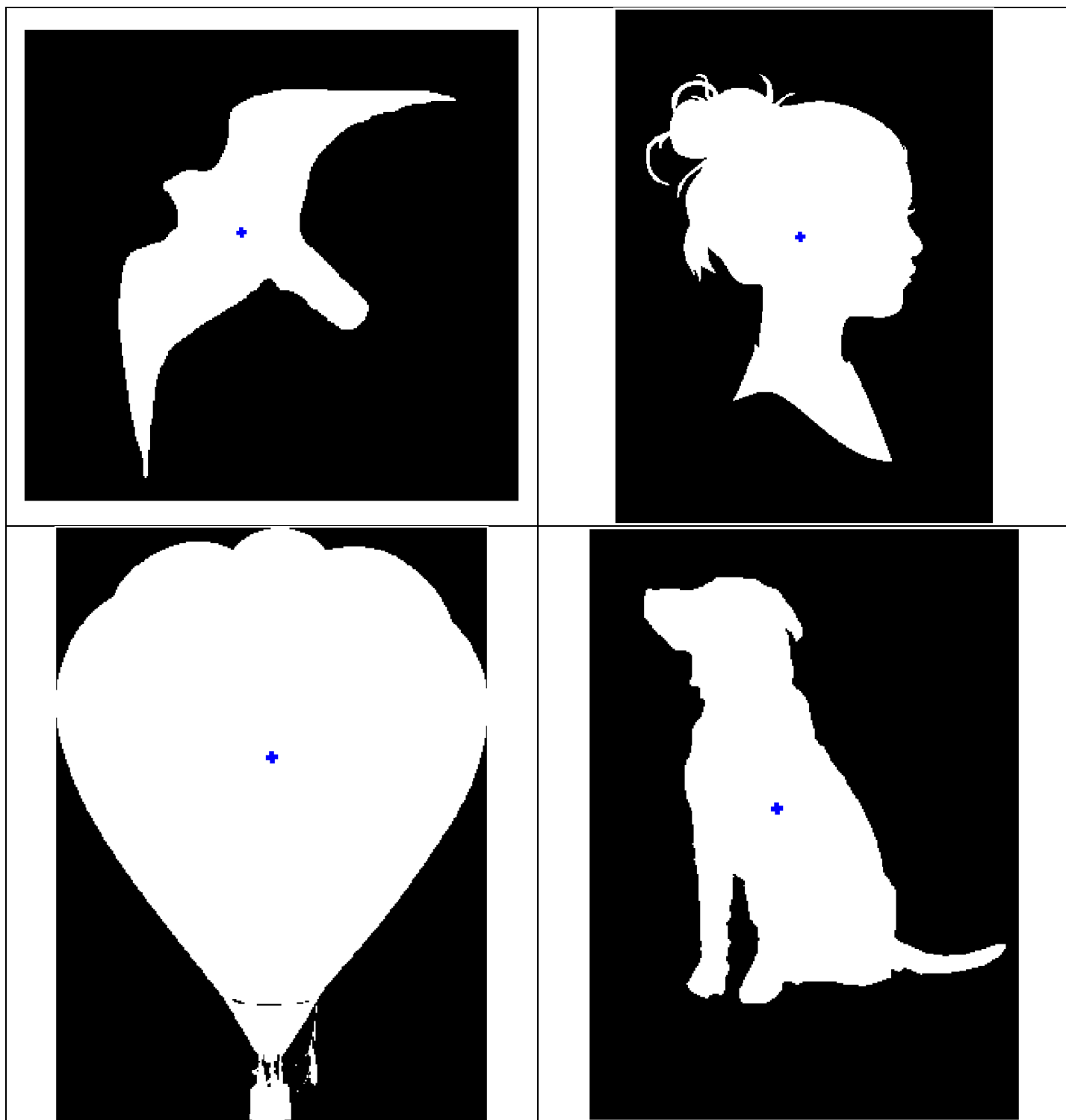
الگوریتم **Thining** در دو نسخه تهیه شده است؛ یکی به صورت دستی و مطابق الگوریتم بالا نوشته شده است و دیگری از تابع آماده‌ی متلب با نام `bwmorph(imgBinInv,'skel',inf)` و به صورت `bwmorph()` استفاده می‌نماید. نتایج نهائی در ادامه می‌آید که تا حد زیادی مشابه یکدیگرند:

روش دستی		روش مبتنی بر <code>bwmorph(imgBinInv,'skel',inf)</code>	
			
			
			
			

### قسمت ب)

در این قسمت نیز جهت محاسبه‌ی مرکز هر کدام از اجزاء همبند، مانند سوال ۴، از تابع آماده‌ی `regionprops()` استفاده نموده و از آن جایی که مانند تصویر `'hotballoon.jpg'` بیشتر از یک جزء همبند در تصویر موجود است، مرکز آن جزء همبندی که مساحت آن از همه بیشتر است را انتخاب نموده و مطابق آن چه در ادامه می‌آید بر روی تصویر با یک علامت مشخص می‌کنیم:





### قسمت ج)

\*مورد (ج) تکراری و مشابه قسمت (ب) می باشد\*