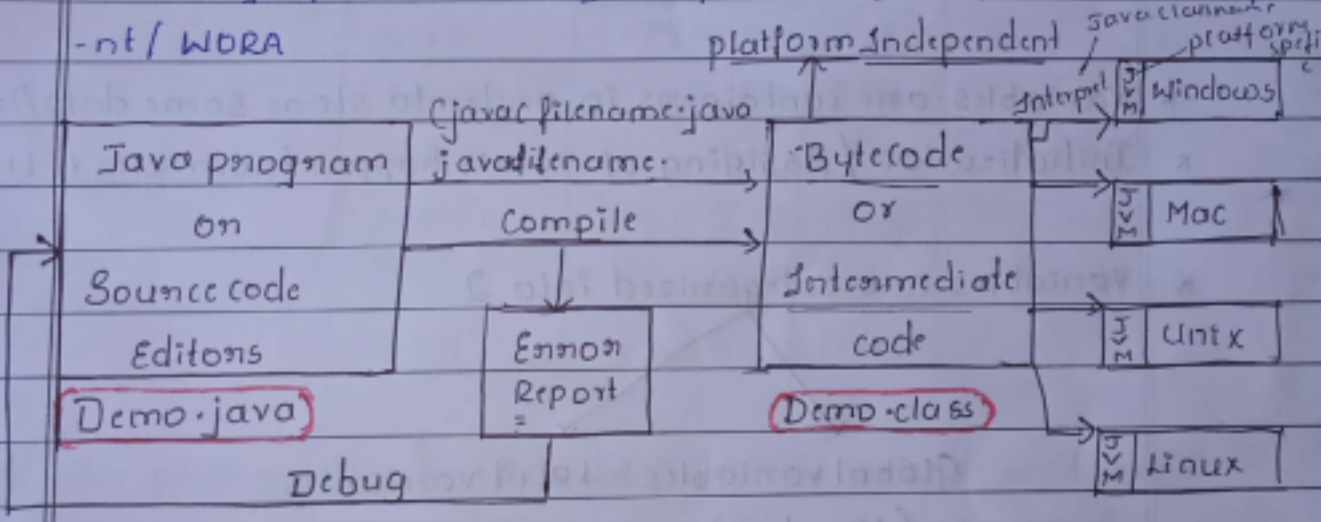
papergrid

Date: 21/01/20

Java

* Java is a high level programming language.
* Programming language is a medium to interact with s/m
* Highlevel language is a language in a normal English i.e Human understandable form
* James Gosling was a person who introduced Java
* The company which started java is SunMicros/m (system)
* Currently Java is owned by oracle.

Working of a Java Program / How is java platform independent / WORA



JVM : Java virtual Machine (Platform dependent)

WORA : Write Once Run Anywhere

Bytecode is an intermediate which is neither low-level nor highlevel language so it uses jvm → (1) convert to machine level (2) Execute line by line

* Firstly we build the java program using editors and save it with the extension .java

* Once we are done developing the program we need to compile it

Scanned with CamScanner

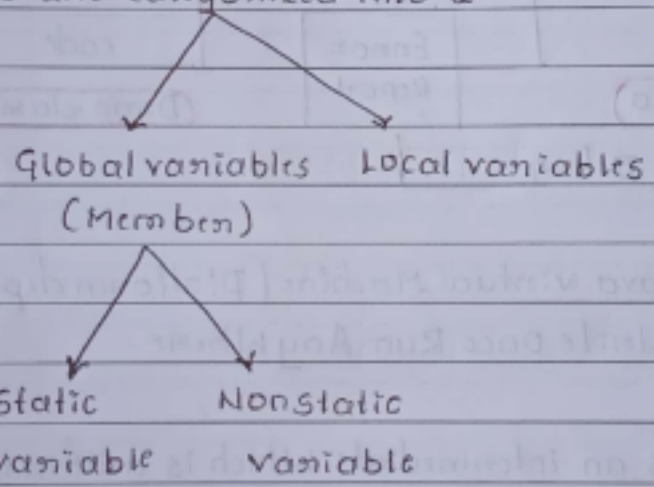
papergrid

Date: / /

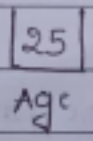
* compilation is a process in onden to check if thene are any ennons in my java pognam on not
* If compilation is unsuccessful we get ennon nepont based on ennon nepont we need to debug the pognam
* If compilation is successful we genenate bytecode which is intenmediate code / platfonm independent code
* Extension of all bytecode is .class
* This bytecode can be executed on all platfonm i.e all operating sh

Vaniables

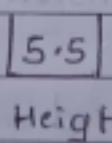
* Vaniables ane containens in onden to stone some data/inform
* Initialization / Assiging of values happen fnom RHS to LHS
* Vaniables ane categonized into 2



Example : Age = 25
LHS RHS



Height = 5.5



V => LHS RHS

Scanned with
CamScanner

DataTypes

* Data types are used to indicate or specify the type of data stored into a variables
* Datatypes are categorized into 2
  1) primitive Datatype
  2) Non primitive Datatype
* In order to store non decimal numeric values we have the following datatypes
* The difference between those datatypes is Memory size

Data types	Memory Size	
	Bytes	Bits
byte	1	8
short	2	16
int	4	32
long	8	64

* In order to store decimal numeric values, we have the following datatypes

Data-T	Memory size	
float	4	32
double	8	64

widely used and default datatype for decimal.

* In order to store true/false we have following datatyp

Data-T	Memory size	
	Bytes	Bits
Boolean	1	8

* In order to store a single character, we make use of char
* char data should be enclosed within 'single quotes' (' ')
* The Memory size of char is 2 bytes = 16bits

Scanned with
CamScanner

Note : All the above mentioned 8 data types are together called as primitive data types

String : It is a dataStr type to store a sequence of character

* String data has to be enclosed within "double quotes"

Note : Java is CaseSensitive Where in lowercase letters are not equivalent to upper case letters/values (a≠A)

22/01/2020

Variable Declaration

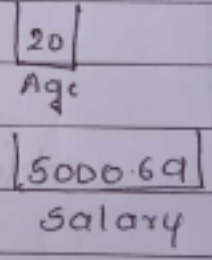
Syntax:

```
datatype VariableName;
int age;
double salary;
```

Variable Initialisation

Syntax :

```
VariableName = value;
age = 20;
salary = 5000.69
```



Variable Declaration & Initialization

Syntax :

```
datatype VariableName = value;
```

Initialization

Declaration

```
boolean x = true;
          false;
```

Scanned with
CamScanner

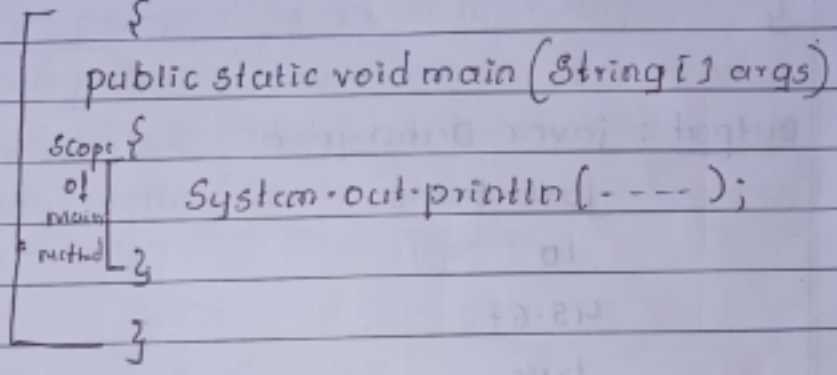
Date: / /

String subject = "java";
= "jsp2020"; (string can store number, character but it should be within " " quotes)
char gender = 'M';

Structure of a java program:

1) Class
2) Main method
3) Print Statement

(i) class: Class className



The filename should be same as classname during file saving className.java

Ex:1)
```
class Firstprogram
{
Public Static Void main (String[] args)
{
System.out.println ("Hello world");
}
}
```

Output: Open cmd
cd desktop
cd java programs
javac Firstprogram.java -(compile)
java Firstprogram -(interprtn)
Hello world!!!

// cd:- change directory to enter to the save files

Scanned with
CamScanner

Ex:2)
```
Class Demo
{
Public static void main (String[] args)
{
System.out.println (10);
System.out.println (45.67);
System.out.println (true);
System.out.println ('2');
System.out.println ("Ijsp@2020");
}
}
```

Output : javac Demo.java

```
java Demo
10
45.67
true
2
Ijsp@2020
```

Ex3) Write a java progrnam to follow the below statement / Scenarios

i) Create class called as student

ii) Define main method

iii) Under main method initialize 2 variables calledas name and age entering those respective values

→
```
Class Student
{
public static void main (String [] args )
{
   String name: "Bhagya";
    int age : 23;
   System.out.println ("name");
   system out. println (age);
}
}
```

O/P

Bhagya

23

Scanned with
CamScanner

papergrid

Date: / /

Note: In java, in order to perform concatenation we make use of '+' operator

Q) class Employee
{
public static void main (String[] args)
{
int id = 101;
String name = "Jerry";
double salary = 123.45;
System.out.println("Employee Id: " +id);
System.out.println("Employee name is: " +name);
System.out.println("Employee Salary: " +salary);
System.out.println(id+" "+name+" "+salary);
}

Output
javac Employee.Java
java Employee
Employee Id: 101
Employee Name is: Jerry
Employee Salary: 123.45
101 Jerry 123.45

23/1/2020

Operators

1) Arithmetic Operators
2) Assignment Operators
3) Relational / Conditional / comparsion Operators
4) Logical Operators
5) Unary Operators

Scanned with
CamScanner

1) Arithmetic Operators

+ : Addition
− : Substraction
* : Multiplication
/ : Division
% : Modulus

5 → divisir
2 | 10
10
0 → Modulus

Ex: 10/2 = 5 10%2 = 0

Example:

```
Class ArithmeticOperators
{
public static void main (String[] args)
  {
     int x=10;
     int y=20;
     int sum = x+y;
     int diff = x-y;
 System.out.println("Sum = "+sum);
 System.out.println(" Difference is " + diff);
 System.out.println (y*5);
 System.out.println (30/3);
 System.out.println (30%3);
```

Output:-

```
Sum = 30
Difference is = -60
100
10
0
```

Scanned with CamScanner

Date: / /

2) Assignment operators

=

+=

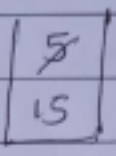
-=

*=

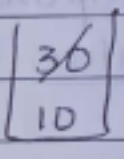
/=

%=

```
int a=5;
   a= a+10   or   a+=10
   a=15
```




```
int x=30
   x-=20
   x=x-20
   x=30-20
   x=10
```



Ex: Class Assignment Operator

```
{
public static void main (String[] args)
 {
  int x=10;
  System.out.println("value of x is :" +x);
     x+=20;
  System.out.println("value of x is :" +x);
  System.out.println("=====");
  int a= 6
  system.out.println("value of a is:" +a);
     a*=5;
  system.out.println("value of a is:" +a);
 }
}
```

Olp

Scanned with
CamScanner

Pathname → cmd
= Assigning
== comparision

papergrid

Date: / /

Output

Value of x is 10
value of x is 30
= = = = = =
value of a is 6
value of a is 30

3) Relational / conditional / comparsion Operators

< : lesser than
> : Greater than
<= : lesser than or equal to
>= : Greater than or equal to
== : Equals to
!= : not equal to

Note : Comparsion Operators will always return boolean values i.e (true/false)

Ex: Class comparsion operator

```
{
  public static void main (String[] args)
      {
          int x=10;
          int y=20;

          boolean result1 = x<y;
          boolean result2 = y>x;
System.out.println(result1 + "\t" + result2);
System.out.println("= = = = = =");
System.out.println(x<=10);
System.out.println(3>=4);
System.out.println("- - - - - -");
System.out.println(x==10);
System.out.println(y==30);
System.out.println("- - - - - -");
System.out.println(x!=10);
System.out.println(y!=30);
```

O/P

true true
= = = = = =
true
false
- - - - - -
true
false
- - - - - -
false
true

Scanned with
CamScanner

4) Logical operators

AND => &&
OR => ||
NOT => !

return => boolean { true = 1, false = 0 }

AND		&&
T	T	T
T	F	F
F	T	F
F	F	F

OR		||
T	T	T
T	F	T
F	T	T
F	F	F

NOT	!
T	F
F	T

Ex: Class logical operators

```
{
public static void main (String[] args)
{
 int x=10;
 int y=20;
 boolean result1 = x<y && y>x;
 boolean result2 = x<y && x==1;
System.out.println (result1);
System.out.println (result2);
System.out.println ("---");
System.out.println (1<2||2>10);
System.out.println (2<1||2==3);
System.out.println ("---");
System.out.println (!true);
System.out.println (!false);
System.out.println ("---");
System.out.println (!(1<2));
 }
}
```

O/P

```
true
false
---
true
false
---
false
true
---
false
```

Scanned with
CamScanner

5) Unary Operators

++ (increment by 1)

-- (Decrement by 1)

++ → pre increment
++ → post increment

-- → pre decrement
-- → post decrement

Ex:
```
Class Unary
{
public static void main (String[] args)
  {
    int x = 5;
   System.out.println ("x:" +x);
     x++;
   System.out.println ("x:"+x);
     ++x;
   System.out.println ("x:"+x);
     x--;
   System.out.println ("x:"+x);
     --x;
   System.out.println ("x:"+x);
    }
}
```

Output ⇒
```
x:5
x:6
x:7
x:6
x:5
```

(1)
```
int x= 10;
int y= x++;
```
post increment → First assign, then increment

x [10 → 11] y [10]

(2)
```
int a= 5
int b= ++a;
```
pre increment → First increment, then assign

a [5 → 6] b [6]

Scanned with
CamScanner

Date: / /

(3) int i=2
j=i--;
post-decrement
First assign, then decrement
i [1] j [2]

(4) int p=50;
int q=--p;
pre-decrement
First decrement, then assign
p [50 49] q [49]

(a) int a=5
int b=++a;
pre-increment
First increment, then assign

Ex) class Unary
{
psvm (String[] args)
{
int q=123;
int w=q++;
System.out.println(q+" "+w);

System.out.println("-----");

int o=444;
int p=--o;
System.out.println(o+" "+p);

}
}

Scanned with
CamScanner

Decision / conditional statements

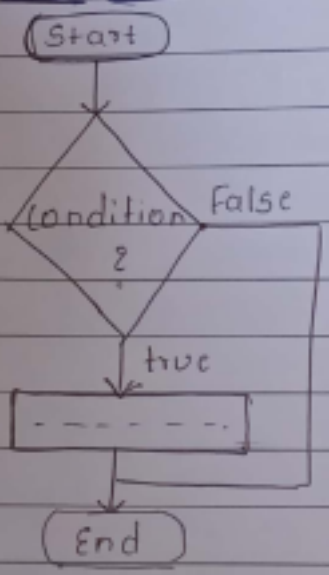
These statements are used to take some decision based on the condition specify. The different decision statements are as follows

1) Simple if
2) if else
3) if else if
4) Nested if
5) Switch

(1) Simple if

Simple if is a decision making statements where in we execute a set of instructions only if the condition is true

Ex: Flow chart



Syntax:

```
if (condition)
{
  statement 1;
  - - - - -
  statement n;
}
```

Set of instructions

Ex:

```
class SimpleifDemo
{
psvm (String[] args)
{
 System.out.println ("START");
}
    int a=10;
    int b=10;
```

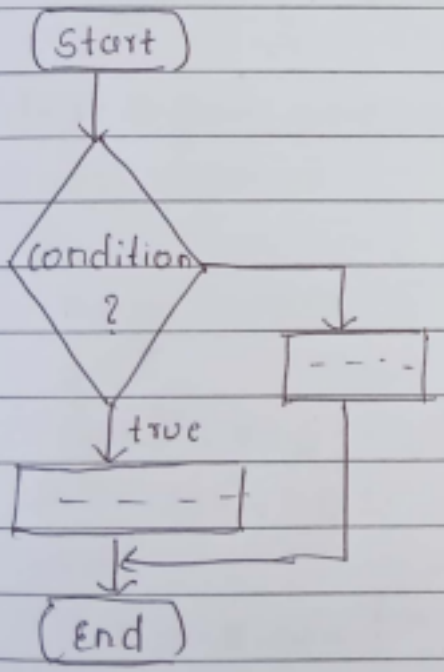
Scanned with
CamScanner

```
if (a<=b)
{
system.out.println (a+" is less than or equal to" +b);
}
system.out.println ("End");
}
}
```

(2) if else :-

x If else is a decision making statement where in, if the condition is true we execute if block. Otherwise if the condition is false we execute else block.

flow chart



Syntax :

```
if (condition)
{
statement 1;
- - - -
statement n;
}
else
{
- - - -
}
```

set of instructions

Scanned with CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

Scanned with CamScanner

Scanned with CamScanner

Scanned with CamScanner

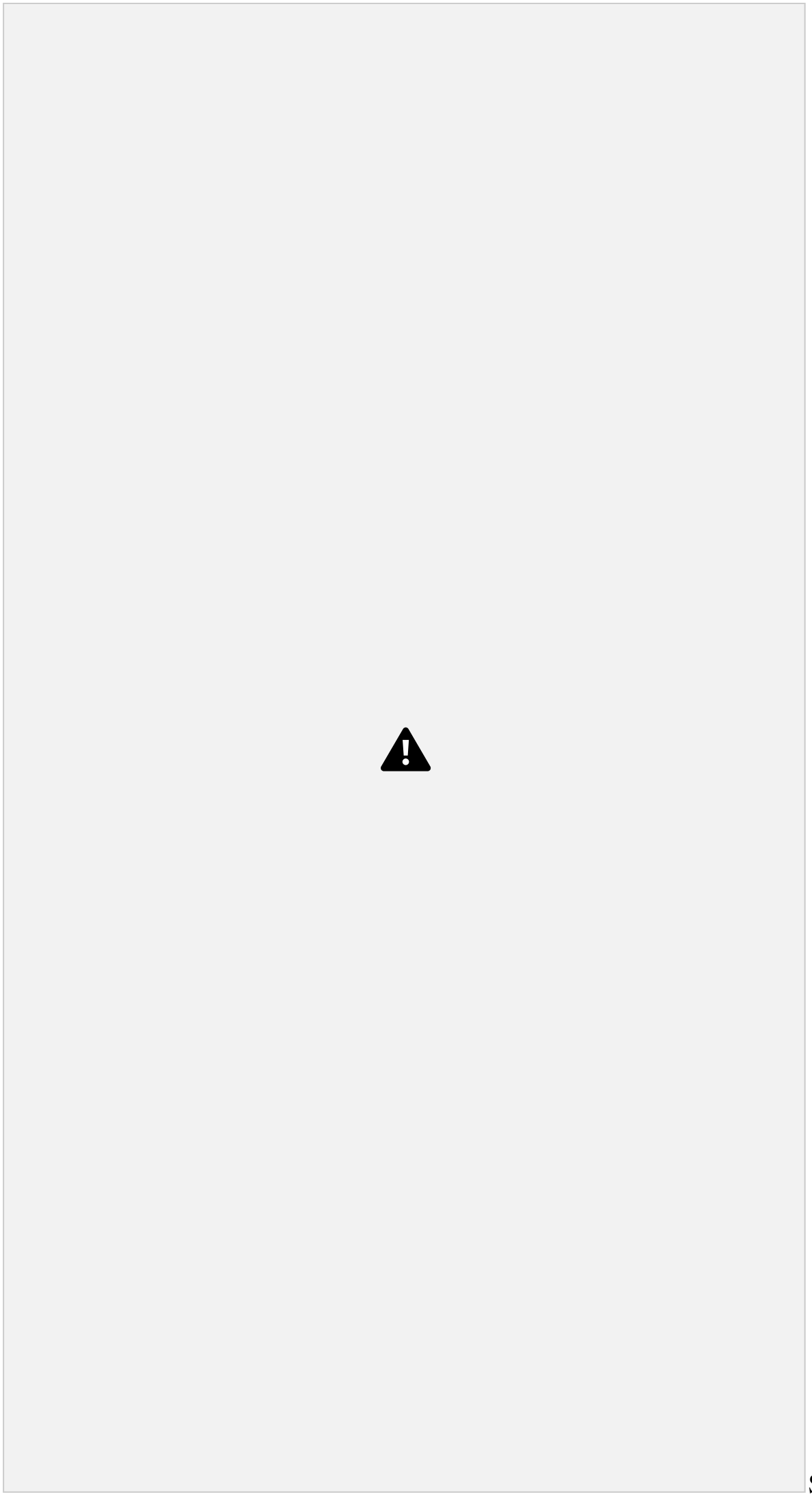
Scanned with CamScanner

Scanned with CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

Scanned with CamScanner


Scanned

with CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

Scanned with CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

Scanned with CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

Scanned with CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

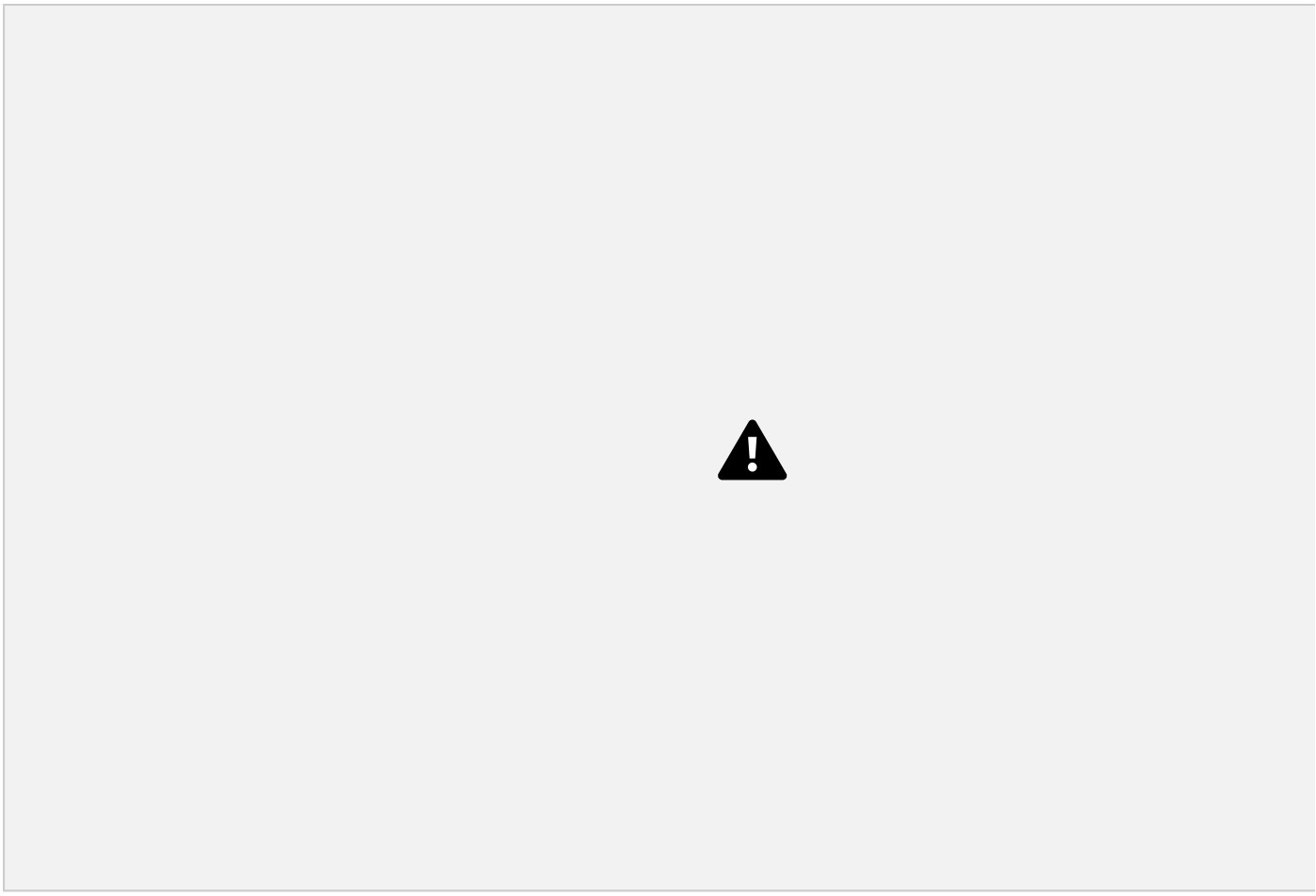
Scanned with CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

Scanned with
CamScanner



Scanned with CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

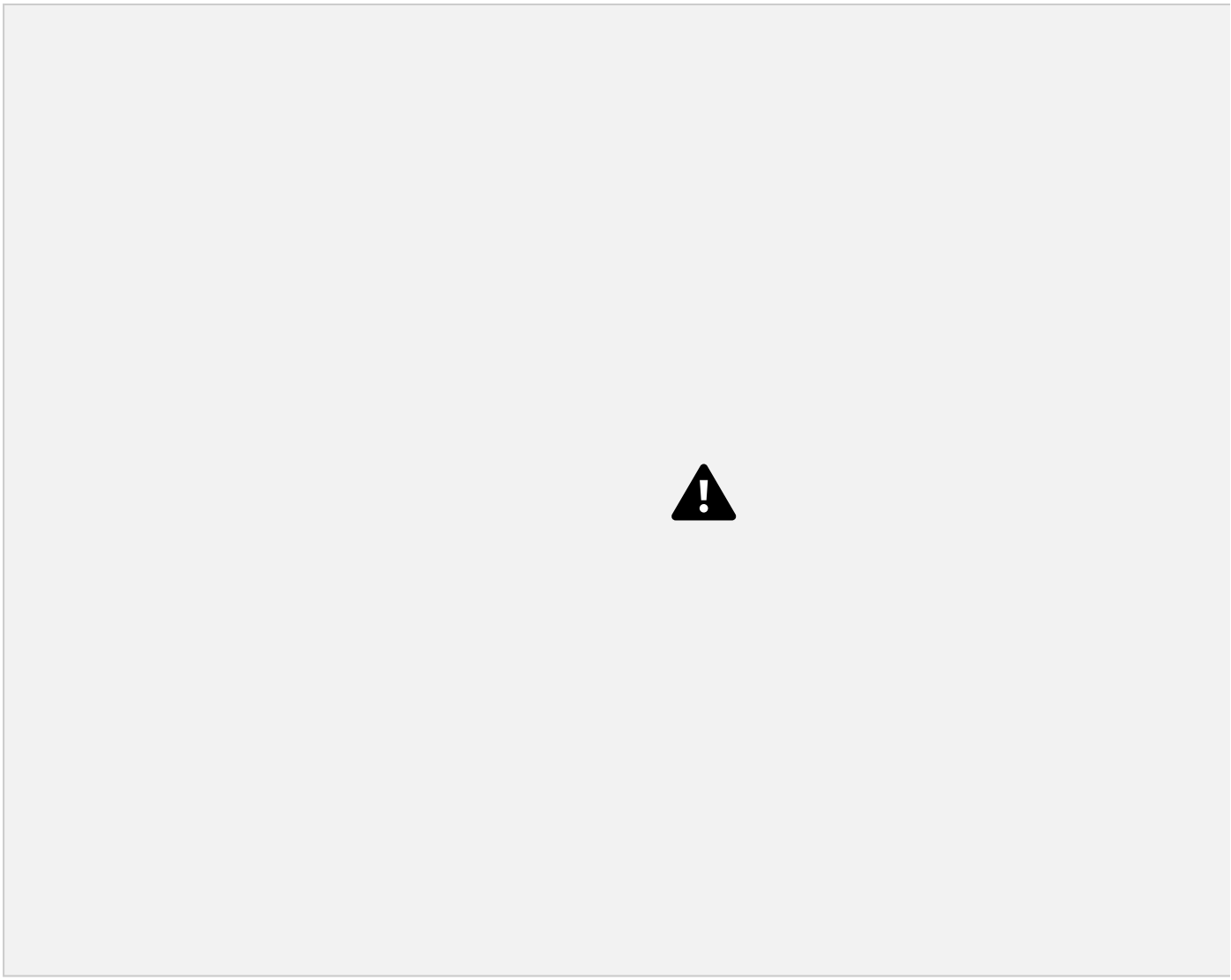
Scanned with
CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

Scanned with CamScanner

Scanned with CamScanner



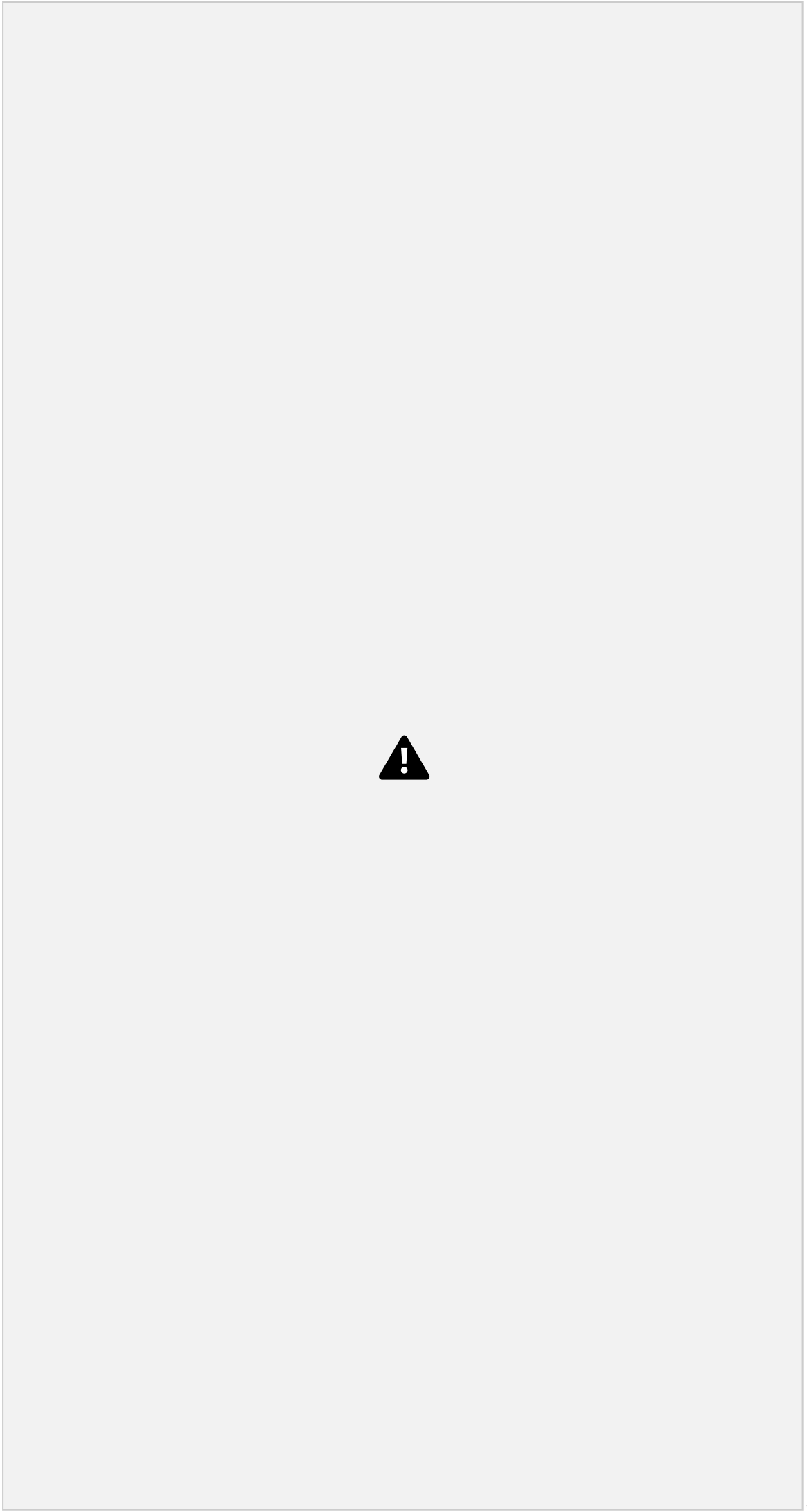
Scanned with CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

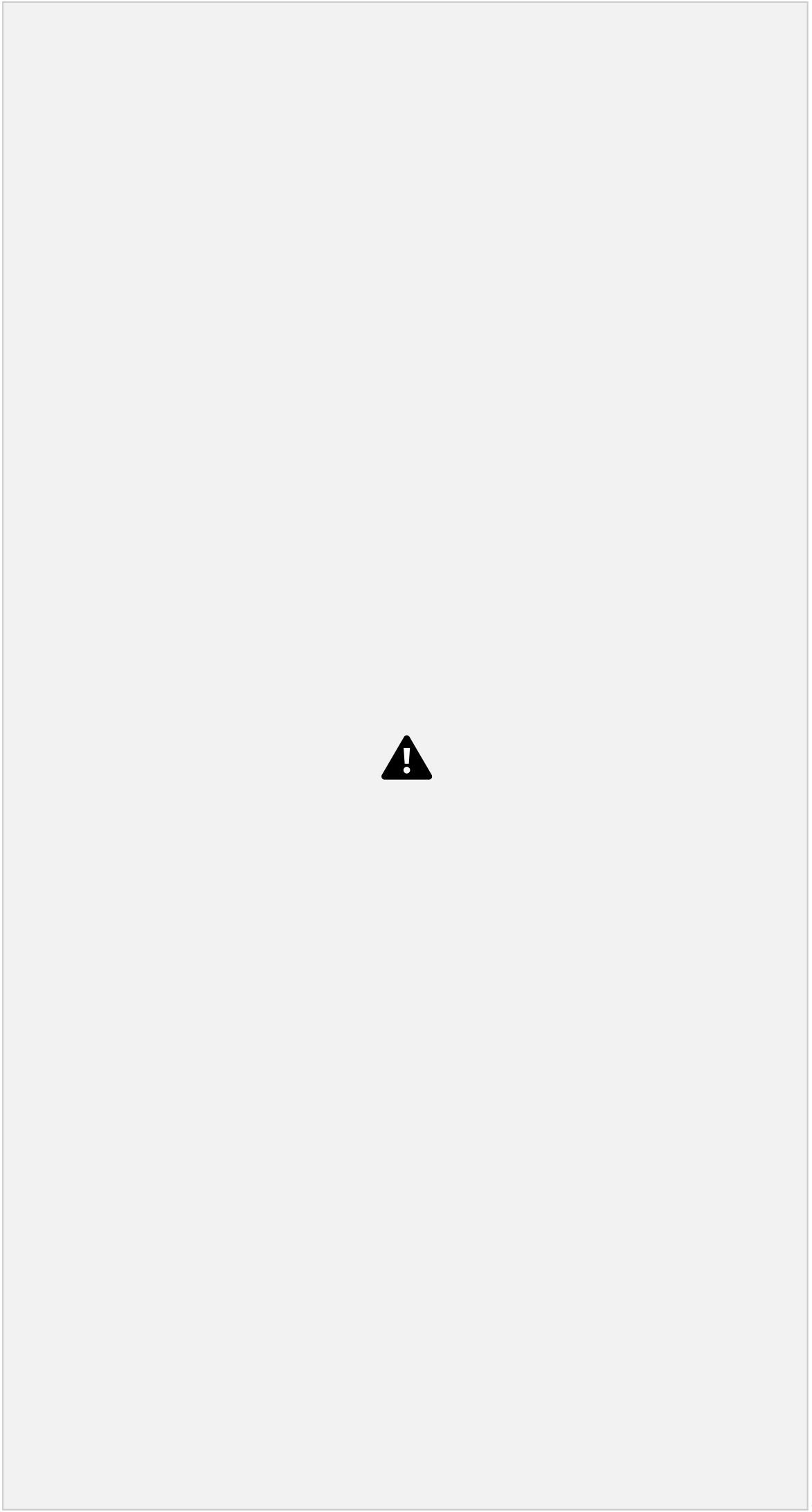

Scanned

with CamScanner

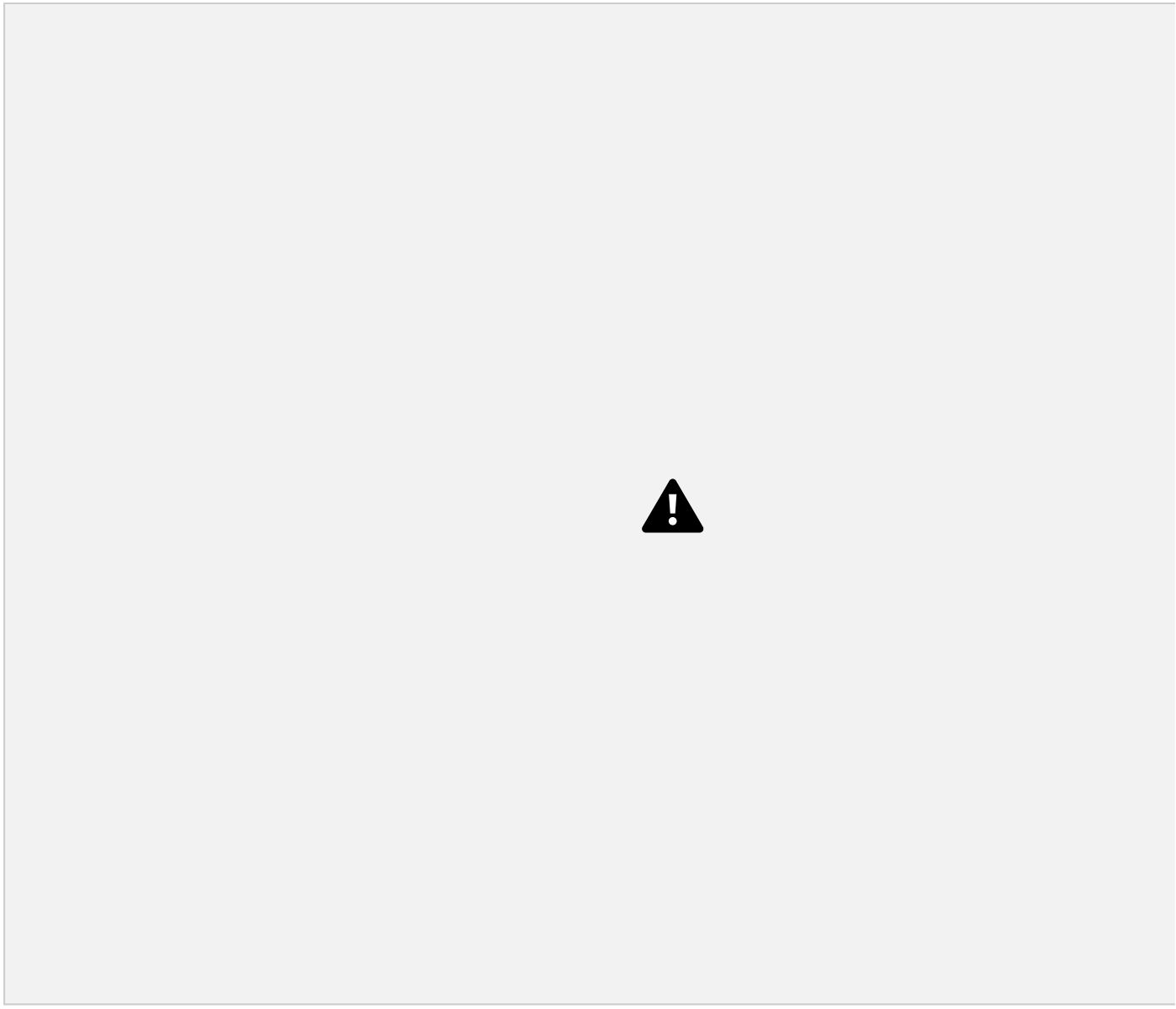
Scanned with
CamScanner

Scanned with
CamScanner

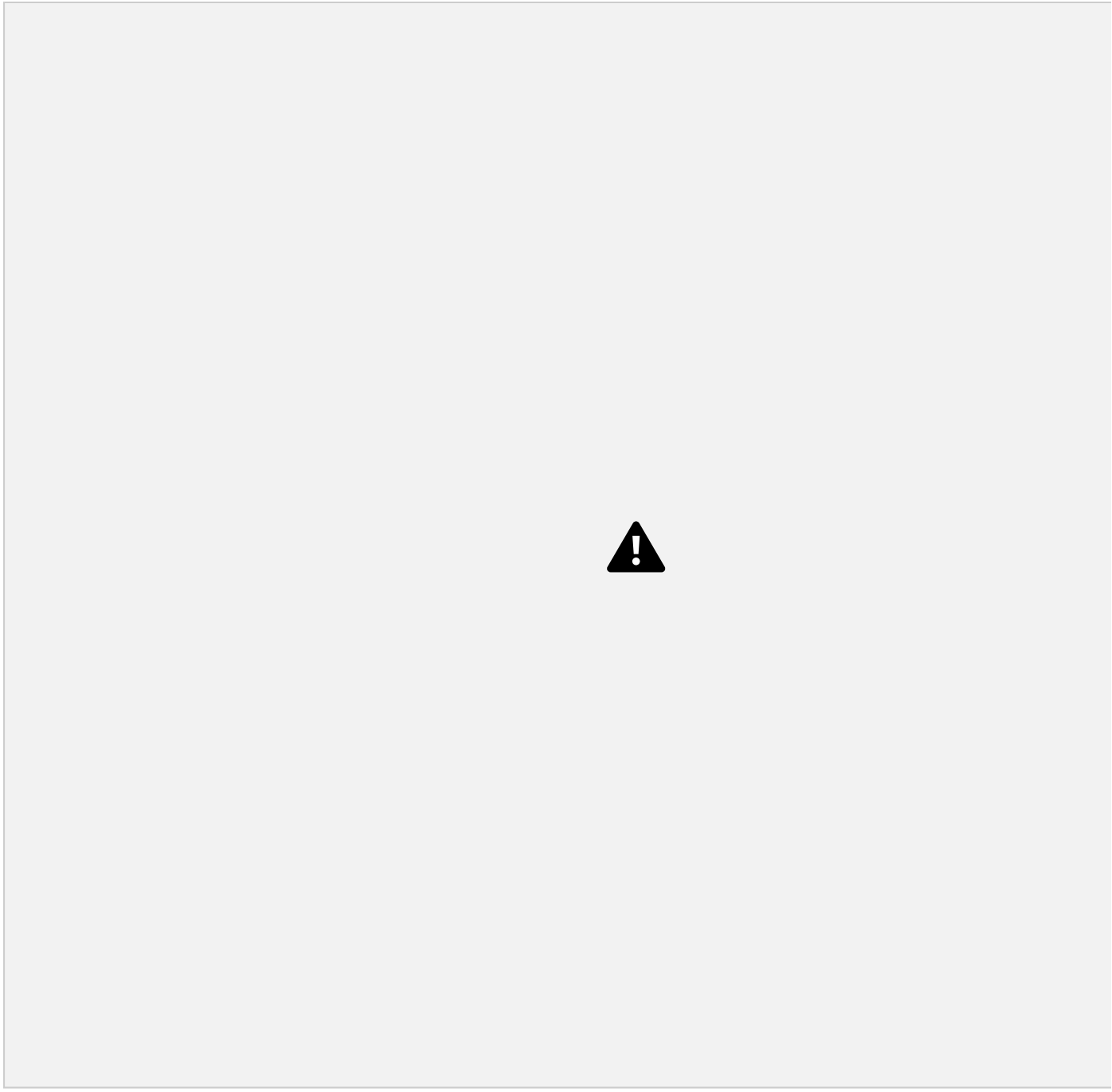
Scanned with
CamScanner


Scanned

with CamScanner



Scanned with CamScanner



Scanned with CamScanner

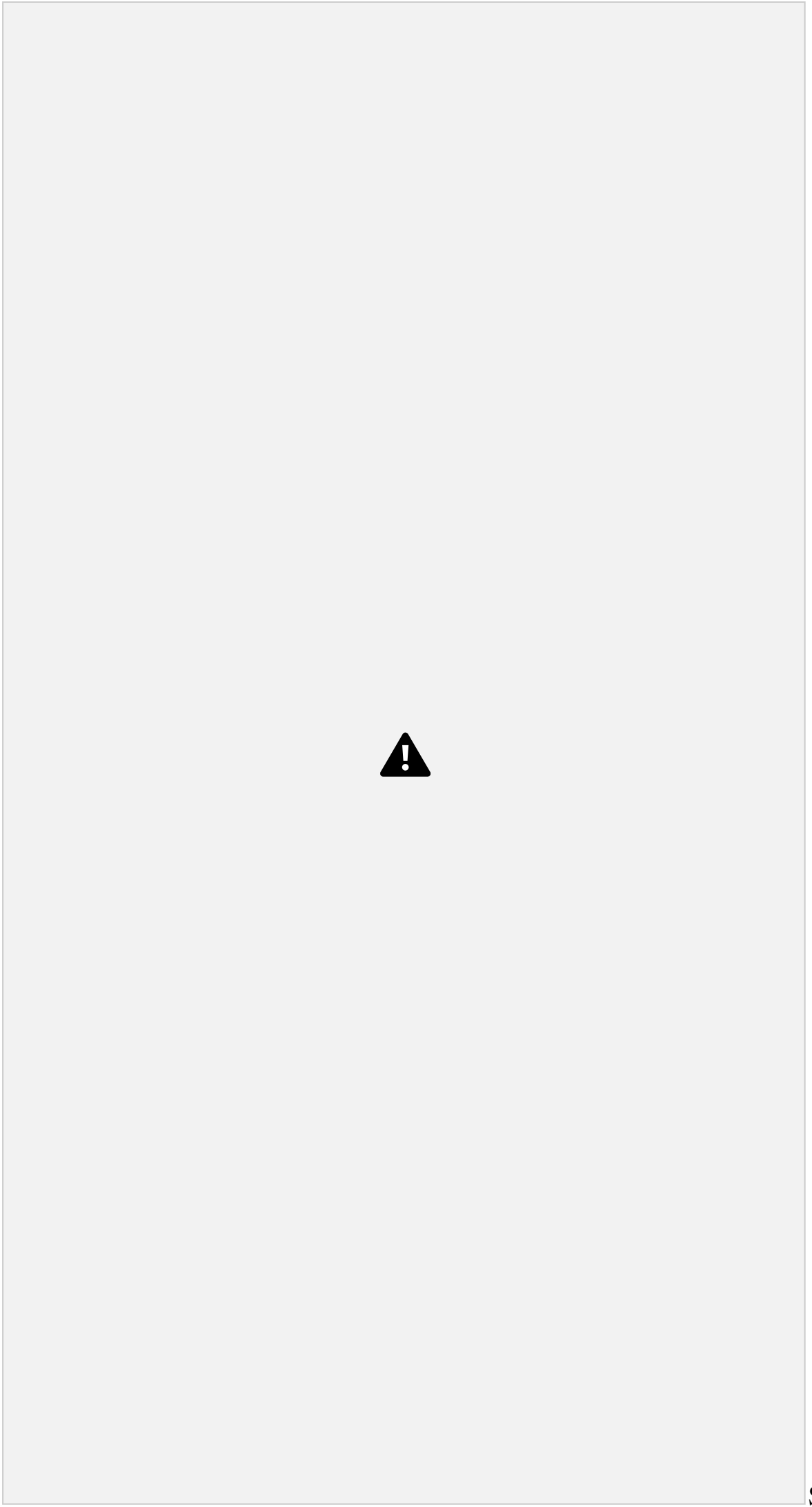
Scanned with CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

Scanned with
CamScanner

Scanned with
CamScanner


Scanned

with CamScanner

Scanned with CamScanner

Scanned with CamScanner

Scanned with
CamScanner