

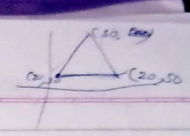
② go-Font.

```

import java.applet.*;
import java.awt.*;
public class font extends Applet
{
    String msg;
    Font f;
    public void init()
    {
        f = new Font("Times New Roman",
                     Font.BOLD, 30);
        setFont(f);
    }
    public void paint(Graphics g)
    {
        g.drawString("hello", 200, 200);
        f = g.getFont();
        msg = f.getName();
        g.drawString("FONT NAME IS " + msg,
                     300, 300);
        g.drawPolygon(x, y, num);
        g.fillPolygon(x, y, num);
    }
}
<applet code="font.java" width=500
height=500 ></applet>

```

③ Polygon



```

import java.applet.*;
import java.awt.*;
public class polygon extends
    Applet
{
    int num = 3;
    int x[] = { 50, 100, 250, 100 };
    int y[] = { 100, 280, 450, 290 };
    public void init()
    {
        public void paint(Graphics g)
        {
            g.drawPolygon(x, y, num);
            g.fillPolygon(x, y, num);
        }
    }
}
<applet code="polygon.java"
width=500 height=500 ></applet>

```

④ Palindrome:-

```

import java.util.*;
class palind
{
    public static void main (String args[])
    {
        int res = 0;
        Scanner sc = new Scanner (System.in);
        num = sc.nextInt();
        num1 = num;
        while (num > 0)
        {
            remainder = num % 10;
            res = res * 10 + remainder;
            num = num / 10;
        }
        System.out.println("Palindrome is " + res);
        if (num1 == res)
            System.out.println("Entered number is palindrome.");
        else
            System.out.println("Entered number is not a palindrome.");
    }
}

```

⑤ Prime Number prg:-

```

import java.util.*;
class prime
{
    public static void main (String args[])
    {
        int num;
        boolean flag = true;
        Scanner sc = new Scanner (System.in);
        num = sc.nextInt();
        for (int i = 2; i <= num / 2; i++)
        {
            if (num % i == 0)
            {
                flag = false;
                break;
            }
        }
        System.out.println("not prime");
        if (flag == true)
            System.out.println("prime num");
        else
            System.out.println("Not prime");
    }
}

```


★ Fibonacci Series

class fibo

{

public static void main (String args[])

{

int num1 = 0, num2 = 1, ans;

System.out.println(num1);

while (num2 < 100)

{

ans = num1 + num2;

num1 = num2;

num2 = ans;

}

}

}

★ Sum of digit of number entered by the user.

import java.util.*;

class Sum

{

public static void main (String args[])

{

Scanner res = new Scanner(System.in);

$$\begin{array}{r} 10 \overline{) 123} \\ \underline{10} \\ 23 \\ \underline{20} \\ 30 \\ \underline{30} \\ 0 \end{array}$$

$$\begin{array}{r} 10 \overline{) 123} \\ \underline{10} \\ 23 \\ \underline{20} \\ 30 \\ \underline{30} \\ 0 \end{array}$$

$$\begin{array}{r} 10 \overline{) 123} \\ \underline{10} \\ 23 \\ \underline{20} \\ 30 \\ \underline{30} \\ 0 \end{array}$$

$$\begin{array}{r} 10 \overline{) 123} \\ \underline{10} \\ 23 \\ \underline{20} \\ 30 \\ \underline{30} \\ 0 \end{array}$$

$$\begin{array}{r} 10 \overline{) 123} \\ \underline{10} \\ 23 \\ \underline{20} \\ 30 \\ \underline{30} \\ 0 \end{array}$$



SUMMER- 18 EXAMINATION

Name: Java Programming

Model Answer

Subject Code:

17515

Applets are small applications that are accessed on an Internet server, transported over the Internet, automatically installed, and run as part of a web document. The applet states include:

- Born or initialization state
- Running state
- Idle state
- Dead or destroyed state

Initialization state: Applet enters the initialization state when it is first loaded. This is done by calling the `init()` method of Applet class. At this stage the following can be done:

- Create objects needed by the applet
- Set up initial values
- Load images or fonts
- Set up colors

Initialization happens only once in the life time of an applet.

```
public void init()
{
//implementation
}
```

Running state: applet enters the running state when the system calls the `start()` method of Applet class. This occurs automatically after the applet is initialized. `start()` can also be called if the applet is already in idle state. `start()` may be called more than once. `start()` method may be overridden to create a thread to control the applet.

```
public void start()
{
//implementation
}
```

Idle or stopped state: an applet becomes idle when it is stopped from running. Stopping occurs automatically when the user leaves the page containing the currently running applet. `stop()` method may be overridden to terminate the thread used to run the applet.

```
public void stop()
{
//implementation
}
```

Dead state: an applet is dead when it is removed from memory. This occurs automatically by invoking the `destroy` method when we quit the browser. Destroying stage occurs only once in the lifetime of an applet. `destroy()` method may be overridden to clean up resources like threads.

```
public void destroy()
{
//implementation
}
```



SUMMER- 18 EXAMINATION	
Subject Name: Java Programming	Subject Code: 17519
<u>Model Answer</u>	

<pre>public class Test { public static void main(String args[]) { int i = 100; long l = i; // no explicit type casting require float f = l; // no explicit type casting required System.out.println ("Int value " + i); System.out.println ("Long value " + l); System.out.println ("Float value " + f); } }</pre>	<p>Output:</p> <p>Int value 100</p> <p>Long value 100</p> <p>Float value 100.0</p>
---	--

2. Narrowing or Explicit type casting

- When you are assigning a larger type value to a variable of smaller type. Then you need to perform explicit type casting.

<pre>public class Test { public static void main(String args[]) { double d = 100.04; long l = (long) d; // explicit type casting required int i = (int) l; // explicit type casting required System.out.println ("Double value " + d); System.out.println ("Long value " + l); System.out.println ("Int value " + i); } }</pre>	<p>Output:</p> <p>Double value 100.04</p> <p>Long value 100</p> <p>Int value 100</p>
--	--



Unit	Unit Outcomes (UOs) (in cognitive domain)	Topics and Sub-topics
	settings.	size, style, font methods: getFamily(), getFont(), getFontname(), getSize(), getStyle(), getAllFonts() and get available font family name() of the graphics environment class.
Unit –VI Managing Input /Output/ Files in Java	6a. Use I/O stream classes in a program to solve the given problem. 6b. Write programs for reading and writing character streams to and from the given files. 6c. Write programs for reading and writing bytes to and from the given files. 6d. Write program to demonstrate use of primitive Data types with the specified stream.	6.1 Introduction and Concept of Streams. 6.2 Stream Classes. 6.3 Byte Stream Classes: Input Stream Classes, Output Stream Classes. 6.4 Character Stream Classes, Using streams. 6.5 Using File Class: I/O Exceptions, Creation of Files, Reading/Writing characters, Reading/Writing Bytes, Handling Primitive Data types.

Note: To attain the COs and competency, above listed UOs need to be undertaken to achieve the 'Application Level' of Bloom's 'Cognitive Domain Taxonomy'

9. SUGGESTED SPECIFICATION TABLE FOR QUESTION PAPER DESIGN

Unit No.	Unit Title	Teaching Hours	Distribution of Theory Marks			
			R Level	U Level	A Level	Total Marks
I	Basic Syntactical constructs in Java	06	02	04	04	10
II	Derived Syntactical Constructs in Java	10	02	06	10	18
III	Inheritance, Interface and Package	10	02	04	06	12
IV	Exception Handling and Multithreading	08	02	04	06	12
V	Java Applets and Graphics Programming	08	02	04	04	10
VI	Managing Input/Output/Files in Java	06	02	02	04	08
Total		48	12	24	34	70

Legends: R=Remember, U=Understand, A=Apply and above (Bloom's Revised taxonomy)

Note: This specification table provides general guidelines to assist students for their learning and to teachers to teach and assess students with respect to attainment of UOs. The actual distribution of marks at different taxonomy levels (of R, U and A) in the question paper may vary from above table

10. SUGGESTED STUDENT ACTIVITIES

Other than the classroom and laboratory learning, following are the suggested student-related **co-curricular** activities which can be undertaken to accelerate the attainment of the various outcomes in this course: Students should conduct following activities in group and prepare reports of about 5 pages for each activity, also collect/record physical evidences for their (student's) portfolio which will be useful for their placement interviews:



- Prepare journals based on practical performed in laboratory.
- Follow coding standards.
- Develop variety of programs to improve the logical skills.
- Develop Application oriented real world programs.
- Prepare power point presentation or animation for understanding different Object

...polling there are three in-built methods that take part in inter-thread communication -

notify()	If a particular thread is in the sleep mode then that thread can be resumed using the notify call.
notifyall()	This method resumes all the threads that are in suspended state.
wait()	The calling thread can be send into a sleep mode.

4.19 Deadlock

Deadlock is a situation in which two or more threads are waiting for object lock which is hold by some another thread.

For example - Consider thread A is waiting for object obj2 which is locked by thread B and thread B is waiting for object obj1 which is locked by thread A. This situation is described as deadlock.

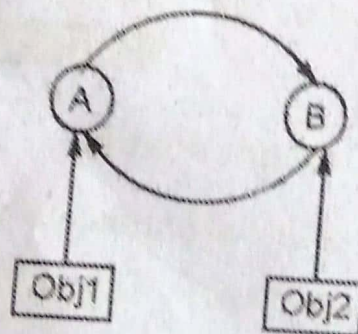


Fig. 4.19.1 : Deadlock

Following Java program shows the deadlock situation

Java Program[DeadLockDemo.java]

```
public class DeadLockDemo
{
```



Board Questions

1. Explain thread priority and method to get and set priority values.

MSBTE : Summer 15, Marks 4

2. What is thread priority ? How thread priority are set and changed ? Explain with example.

MSBTE : Summer 16, Marks 8

3. What is thread priority ? Write default priority values and methods to change them.

MSBTE : Summer 17, Marks 4

4.17 Synchronization

MSBTE.: Winter 16, Marks 4

- When two or more threads need to access shared memory, then there is some way to ensure that the access to the resource will be by only one thread at a time. The process of ensuring one access at a time by one thread is called synchronization. The synchronization is the concept which is based on monitor. Monitor is used as mutually exclusive lock or mutex. When a thread owns this monitor at a time then

e) Write a program to count number of words from a text file.

Q.5) Attempt any TWO of the following.

12 Marks

a) Write a step to declare and define two and three dimensional arrays of a class.

```
result = (num1 > num2) ? num1 : num2  
System.out.println("Largest is!" +  
    result);
```

```
}
```

```
}
```

```
student (int r, String n)
```

```
{
```

```
rollno = r;
```

```
name = n;
```

```
}
```

```
Student (int r)
```

```
{
```

```
rollno = r;
```

```
}
```

```
void display ()
```

```
{
```

```
System.out.println("Rollno is: " + rollno);
```

```
System.out.println("Name is: " +
```


class sdemos

{

public static void main
(String args[])

{

Student s = new Student(10, "RAJ");

s.display();

Student s = new Student(20, "ABHI");

s.display();

}

}

⊛ Write a program to find largest between two numbers using ">:" operator.

→

import java.util.*;

class large

{

public static void main

(String args[])

{

int num1, num2, result;

Scanner sc = new Scanner

(System.in);

System.out.println("Enter num.
& num2:");

num1 = sc.nextInt();

num2 = sc.nextInt();

- Java allows its programmers to initialize a variable at run time also. Initializing a variable at run time is called dynamic initialization.

```
Double sr = Math.sqrt(100);
```

Syllabus Topic : Array and Strings

1.11 Array and Strings

- Array is a group of elements with similar data types.
- Array of characters is known as String.
- This topic we are going to discuss in detail in next chapter.

Syllabus Topic : Scope of Variable

1.12 Scope of Variable

→ (MSBTE - W-16, S-18)

Q. 1.12.1 What is scope of variable? Give example of class variable, instance variable and local variable.

(Refer section 1.12)

W-16, 4 Marks

Q. 1.12.2 State & explain scope of variable with an example.

(Refer section 1.12)

S-18, 4 Marks

- Scope** of a variable refers to; areas or sections of a program in which the variable can be accessible and **lifetime** of a variable refers to how long the variable stays alive in the memory.
- General convention for a variable's scope is, it is accessible only within the block in which it is declared. A block begins with a left curly brace { and ends with a right curly brace }.

- There are three types of variables :

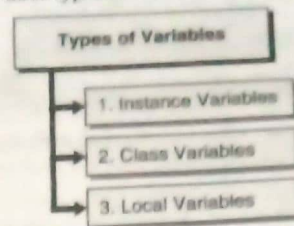


Fig. 1.12.1 : Types of Variables

→ 1. Instance Variables

- A variable which is declared inside a class and outside all the methods and blocks is an instance variable.
- General scope of an instance variable is throughout the class except in static methods.

→ 2. Class Variables

- A variable which is declared inside a class, outside all the blocks and is marked *static* is known as a class variable.
- General scope of a class variable is throughout the class.

→ 3. Local Variables

- All other variables which are not instance and class variables are treated as local variables including the parameters in a method.
- Scope of a local variable is within the block in which it is declared.

Syllabus Topic : Typecasting and Standard Default Values

1.13 Typecasting and Standard Default Values

A) Typecasting

→ (MSBTE - W-14, S-18)

Q. 1.13.1 What do mean by typecasting? When it is needed? (Refer section 1.13)

W-14, 4 Marks

1. Introduction

The purpose of this study is to investigate the effect of the independent variable on the dependent variable. The study is designed to provide a comprehensive overview of the research findings.

2. Literature Review

The literature review covers the existing research on the topic. It highlights the gaps in the current knowledge and identifies the need for the present study. The review also discusses the theoretical framework and the research hypotheses.

3. Methodology

The methodology section describes the research design, data collection methods, and statistical analysis. The study uses a quantitative approach to collect data from a sample of participants. The data is analyzed using statistical software to test the hypotheses.

The results of the study are presented in this section. The findings show a significant positive relationship between the independent variable and the dependent variable. The results are discussed in the context of the existing literature and the theoretical framework.

The conclusion summarizes the main findings of the study and discusses the implications for future research. The study contributes to the understanding of the relationship between the variables and provides a basis for further exploration.

Table 1: Summary of Research Findings

Variable	Mean	Standard Deviation	Range
Independent Variable	1.5	0.5	1.0 - 2.0
Dependent Variable	2.5	0.5	2.0 - 3.0

Fig. 1: Summary of Research Findings

The study found a significant positive relationship between the independent variable and the dependent variable. The results are consistent with the theoretical framework and the research hypotheses.

The study also found that the independent variable has a significant effect on the dependent variable. The results are discussed in the context of the existing literature and the theoretical framework.

The study concludes that the independent variable has a significant positive effect on the dependent variable. The results are discussed in the context of the existing literature and the theoretical framework.

The study also found that the independent variable has a significant effect on the dependent variable. The results are discussed in the context of the existing literature and the theoretical framework.

4. Conclusion

5. References

The study was supported by the following grants: [Grant 1], [Grant 2], [Grant 3].

The study was conducted by the following researchers: [Researcher 1], [Researcher 2], [Researcher 3].

- Java allows its programmers to initialize a variable at run time also. Initializing a variable at run time is called dynamic initialization.

Double d = Math.sqrt(100);

Syllabus Topic : Array and Strings

1.11 Array and Strings

- Array is a group of elements with similar data types.
- Array of characters is known as String.
- This topic we are going to discuss in detail in next chapter.

Syllabus Topic : Scope of Variable

1.12 Scope of Variable

→ (MSBTE - W-16, S-18)

Q. 1.12.1 What is scope of variable? Give example of class variable, instance variable and local variable.

(Refer section 1.12)

W-16, 4 Marks

Q. 1.12.2 State & explain scope of variable with an example.

(Refer section 1.12)

S-18, 4 Marks

- Scope of a variable refers to; areas or sections of a program in which the variable can be accessible and lifetime of a variable refers to how long the variable stays alive in the memory.

- General convention for a variable's scope is, it is accessible only within the block in which it is declared. A block begins with a left curly brace { and ends with a right curly brace }.

- There are three types of variables :

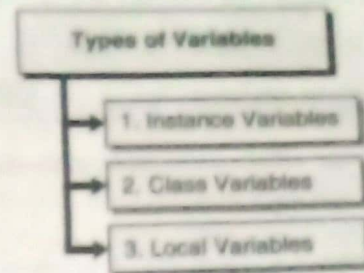


Fig. 1.12.1 : Types of Variables

→ 1. Instance Variables

- A variable which is declared inside a class and outside all the methods and blocks is an instance variable.
- General scope of an instance variable is throughout the class except in static methods.

→ 2. Class Variables

- A variable which is declared inside a class, outside all the blocks and is marked *static* is known as a class variable.
- General scope of a class variable is throughout the class.

→ 3. Local Variables

- All other variables which are not instance and class variables are treated as local variables including the parameters in a method.
- Scope of a local variable is within the block in which it is declared.

Syllabus Topic : Typecasting and Standard Default Values

1.13 Typecasting and Standard Default Values

A) Typecasting

→ (MSBTE - W-14, S-18)

Q. 1.13.1 What do mean by typecasting? When it is needed? (Refer section 1.13)

W-14, 4 Marks

an array is a collection of elements of the same type stored in contiguous memory locations.

Array Declaration

Example: `int arr[10];` // Array of 10 integers

10.1.1 Array Declaration

Array declaration is done at the start of the program.

Example: `int arr[10];` // Array of 10 integers

The array is a collection of elements of the same type stored in contiguous memory locations.

Example: `int arr[10];` // Array of 10 integers

10.1.2 Range of Variables

• Range: 0 to 999

Example: `int arr[10];` // Array of 10 integers
Range: 0 to 999

Example: `int arr[10];` // Array of 10 integers

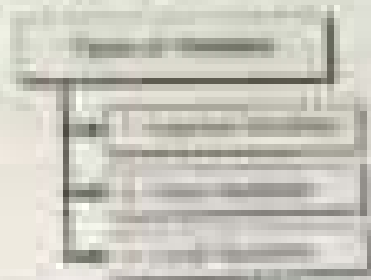
Example: `int arr[10];` // Array of 10 integers
Range: 0 to 999

Example: `int arr[10];` // Array of 10 integers

Range of a variable is the range of values it can store. For example, the range of an integer variable is from -32768 to 32767.

Range of a variable is the range of values it can store. For example, the range of an integer variable is from -32768 to 32767.

Array Declaration



Array Declaration

10.1.1 Array Declaration

Array declaration is done at the start of the program.

Example: `int arr[10];` // Array of 10 integers

10.1.2 Range of Variables

Range of a variable is the range of values it can store. For example, the range of an integer variable is from -32768 to 32767.

Range of a variable is the range of values it can store. For example, the range of an integer variable is from -32768 to 32767.

10.1.3 Array Declaration

Array declaration is done at the start of the program.

Example: `int arr[10];` // Array of 10 integers

Array Declaration, Typing and Storage

10.1 Typing and Storage

10.1.1 Typing

• Range: 0 to 999

Example: `int arr[10];` // Array of 10 integers
Range: 0 to 999

Example: `int arr[10];` // Array of 10 integers

- All these states are represented by five methods means there exists 5 states represented by 5 methods.
 - These methods are automatically called by the browser for smooth execution of the applet.
 - The applet life cycle methods are members of **java.applet.Applet** class except **paint()** method.
 - The **paint()** method is member of **java.awt.Component** class which is an indirect super class of Applet.
- The life cycle of an Applet is as shown in Fig. 5.4.1.

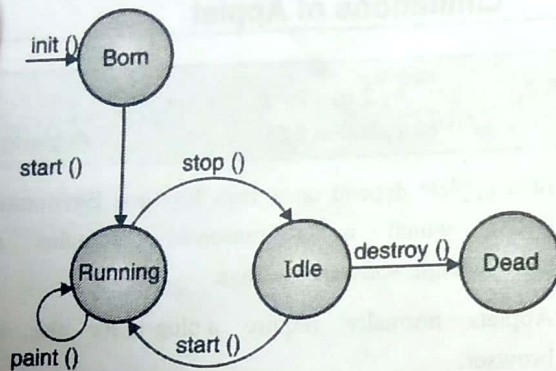


Fig. 5.4.1 : Applet Life Cycle

1. **Init()** : This method signifies the beginning of applet's life cycle. It is used to initialize the Applet. In this method the object of Applet class is created. As this method is called before all the remaining methods, it is used to initialize the variables, instantiate objects, setting background as well as foreground colors in GUI etc.

Invocation : This method is invoked only once at the beginning of an Applet. Consider a website made up of web-pages (applets), when we visit any webpage then first its **init** method is called.

2. **Start()** : After **init()**, the **start()** method is invoked. It starts the execution of Applet. In this state, it is considered that the applet becomes active.

Invocation : This method is repeatedly called in the life of an Applet. This method is invoked whenever user revisits any webpage (Applet). It also executes whenever the applet is restored, maximized or user is shifting from one tab to another tab in the window of browser.

3. **Paint()** : This method is used to create Applet's GUI such as a colored background, drawing different shapes and printing messages etc.

Invocation : This method is called only once by the browser. But user can recall it by using **repaint()** method.

4. **Stop()** : This method is used to stop execution of an applet temporary.

Invocation : This method is repeatedly called in the life of an Applet. Whenever user shifts to another webpage (Applet), this method is invoked. It is invoked when Applet is stopped or browser is minimized.

5. **Destroy()**

This method indicates the end of applet's life cycle. Here user can write the cleanup code. It removes the applet object from memory.

Invocation : This method is called only once by the browser when the applet window is closed or when the browser or the tab of browser containing the webpage is closed.

5.5 Applet Class

Q. 5.5.1 Write a note on Applet Class.
(Refer section 5.5)

(4 Marks)

- Java provides class Applet in the **java.applet** package which contains several methods to handle the functionality of an Applet.
- The hierarchical structure for applets class is follows :

```

java.lang.Object
    ↓
java.awt.Component
    ↓
java.awt.Container
    ↓
java.awt.Panel
    ↓
java.applet.Applet
    
```


REMARK 2. For each $\mathbf{y} \in \mathbb{R}^n$, the vector \mathbf{y}^{opt} is the unique vector in \mathbb{R}^n such that $\mathbf{y}^{\text{opt}} \in \mathcal{C}(\mathbf{y})$ and $\mathbf{y}^{\text{opt}} \in \mathcal{C}(\mathbf{y}^{\text{opt}})$. □

seringall' and 'sag' are not a case of 'lexical
diffusion' as proposed by some. They are not the
same word.

1990	1991	1992	1993	1994	1995	1996	1997	1998	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031	2032	2033	2034	2035	2036	2037	2038	2039	2040	2041	2042	2043	2044	2045	2046	2047	2048	2049	2050	2051	2052	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063	2064	2065	2066	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079	2080	2081	2082	2083	2084	2085	2086	2087	2088	2089	2090	2091	2092	2093	2094	2095	2096	2097	2098	2099	2100	2101	2102	2103	2104	2105	2106	2107	2108	2109	2110	2111	2112	2113	2114	2115	2116	2117	2118	2119	2120	2121	2122	2123	2124	2125	2126	2127	2128	2129	2130	2131	2132	2133	2134	2135	2136	2137	2138	2139	2140	2141	2142	2143	2144	2145	2146	2147	2148	2149	2150	2151	2152	2153	2154	2155	2156	2157	2158	2159	2160	2161	2162	2163	2164	2165	2166	2167	2168	2169	2170	2171	2172	2173	2174	2175	2176	2177	2178	2179	2180	2181	2182	2183	2184	2185	2186	2187	2188	2189	2190	2191	2192	2193	2194	2195	2196	2197	2198	2199	2200	2201	2202	2203	2204	2205	2206	2207	2208	2209	2210	2211	2212	2213	2214	2215	2216	2217	2218	2219	2220	2221	2222	2223	2224	2225	2226	2227	2228	2229	2230	2231	2232	2233	2234	2235	2236	2237	2238	2239	2240	2241	2242	2243	2244	2245	2246	2247	2248	2249	2250	2251	2252	2253	2254	2255	2256	2257	2258	2259	2260	2261	2262	2263	2264	2265	2266	2267	2268	2269	2270	2271	2272	2273	2274	2275	2276	2277	2278	2279	2280	2281	2282	2283	2284	2285	2286	2287	2288	2289	2290	2291	2292	2293	2294	2295	2296	2297	2298	2299	2300	2301	2302	2303	2304	2305	2306	2307	2308	2309	2310	2311	2312	2313	2314	2315	2316	2317	2318	2319	2320	2321	2322	2323	2324	2325	2326	2327	2328	2329	2330	2331	2332	2333	2334	2335	2336	2337	2338	2339	2340	2341	2342	2343	2344	2345	2346	2347	2348	2349	2350	2351	2352	2353	2354	2355	2356	2357	2358	2359	2360	2361	2362	2363	2364	2365	2366	2367	2368	2369	2370	2371	2372	2373	2374	2375	2376	2377	2378	2379	2380	2381	2382	2383	2384	2385	2386	2387	2388	2389	2390	2391	2392	2393	2394	2395	2396	2397	2398</
------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	--------

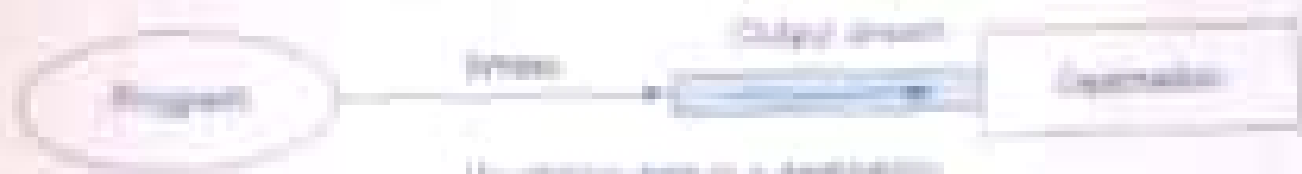
For further information, please contact the author at christian@christian-berthelsen.com.
 The author is not responsible for any errors or omissions in this document.
 The author is not responsible for any damages or losses resulting from the use of this document.
 The author is not responsible for any legal consequences arising from the use of this document.

— *Journal of International Law and Politics*, 2006, 39, 1, 1-24.

11/11/2011 10:00 AM



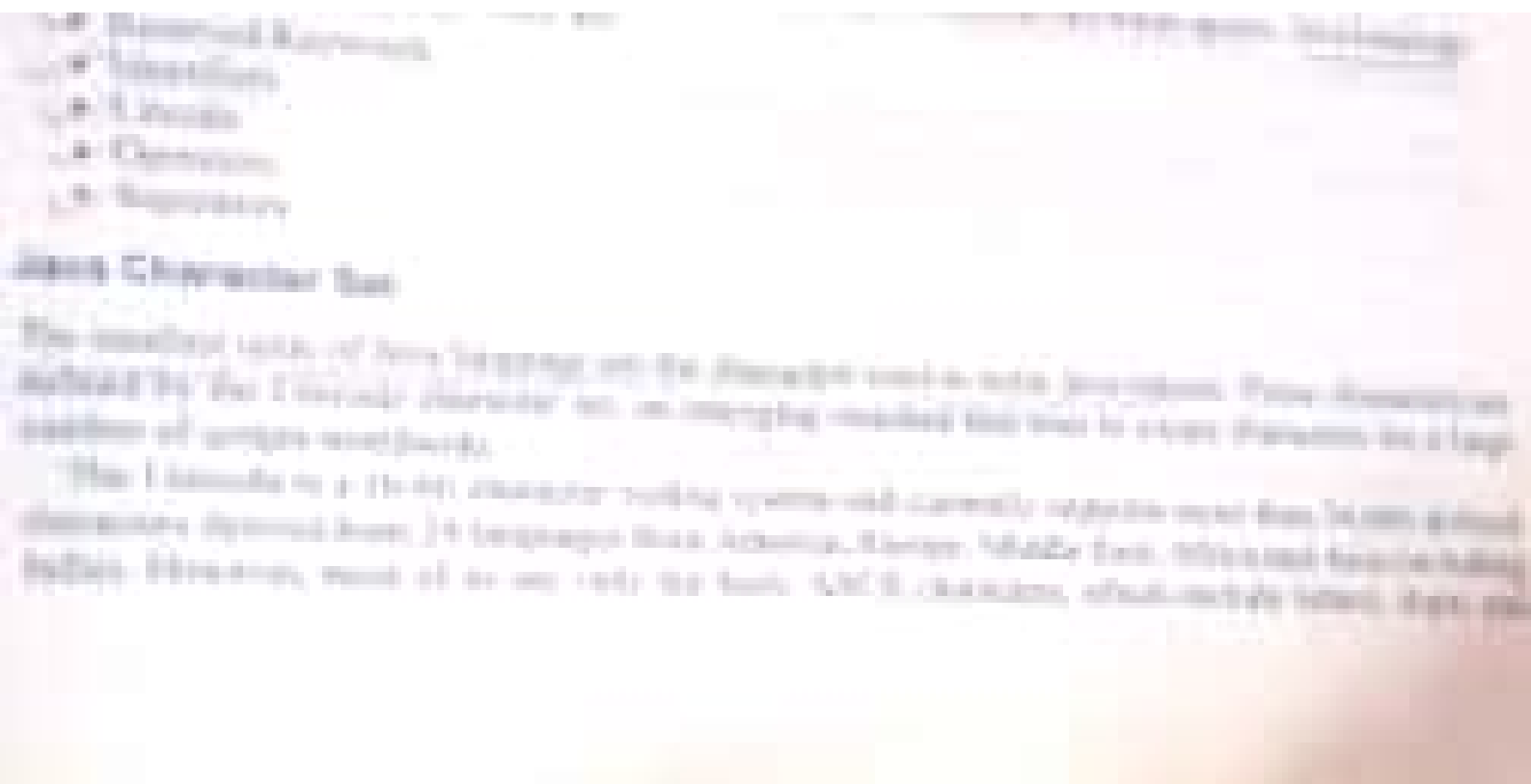
(a) Flowing data into a program



(b) Flowing data out of a program

Fig. 11.4

Using input and output streams



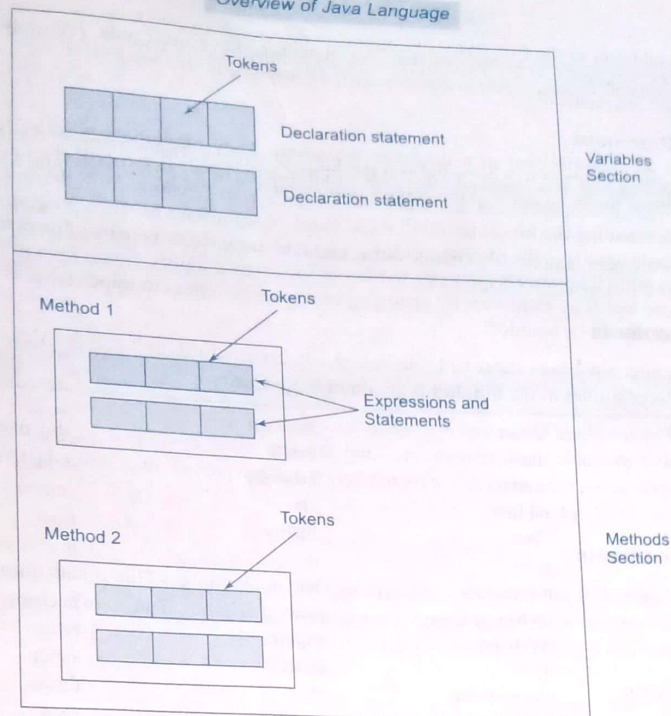


Fig. 3.3 Elements of Java class

In simple terms, a Java program is a collection of tokens, comments and white spaces. Java language includes five types of tokens. They are:

- Reserved Keywords
- Identifiers
- Literals
- Operators
- Separators

Java Character Set

The smallest units of Java language are the characters used to write Java tokens. These characters are defined by the Unicode character set, an emerging standard that tries to create characters for a large number of scripts worldwide.

The Unicode is a 16-bit character coding system and currently supports more than 34,000 defined characters derived from 24 languages from America, Europe, Middle East, Africa and Asia (including India). However, most of us use only the basic ASCII characters, which include letters, digits and