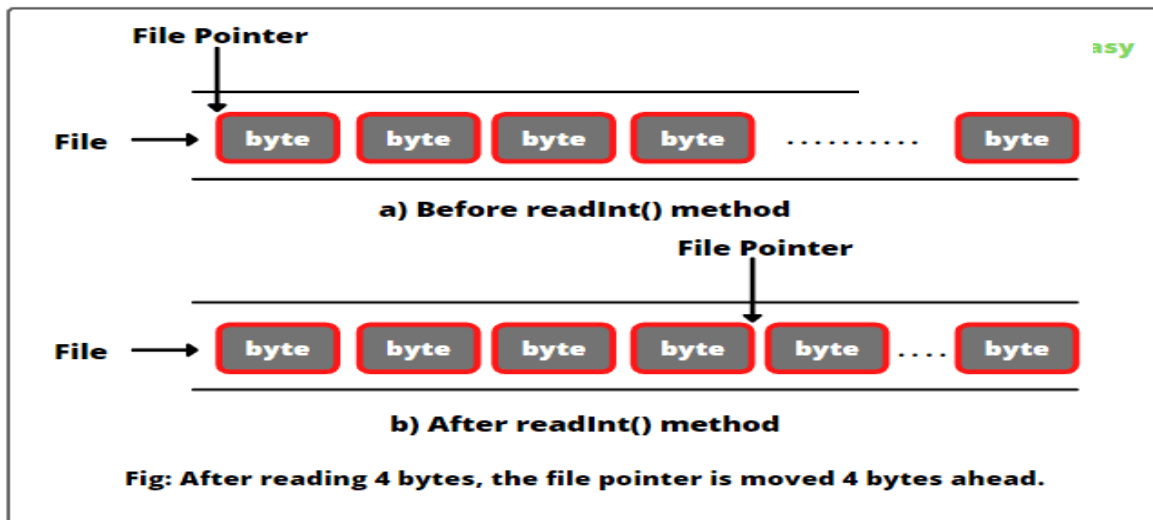


❖ RandomAccessFile class

- Instances of this class support both reading and writing to a random access file.
- A random access file behaves like a large array of bytes stored in the file system.
- A random-access file consists of a sequence of bytes. It supports a special pointer known as file pointer. A file pointer indicates the current position (location) in the file.
- It is positioned at one of these bytes in the file and can be moved to any arbitrary position in the file prior to reading or writing.
- In other words, a read or write operation takes place at the location of the file pointer. The file pointer can be moved using **seek()** method provided by RandomAccessFile class.
- When a file is first created, the file pointer is set to 0, indicating the beginning of the file. When we read or write data to the file using read or write methods, the file pointer moves forward to the next data item (i.e. next byte).
- For example, if we read an **int** value using **readInt()** method from the file, JVM reads **4 bytes** using file pointer, and now the file pointer is **4 bytes ahead** of the previous position, as shown in figure below.



- For a RandomAccessFile raf, the **raf.seek(position)** method **moves** the file pointer to a specified location. **raf.seek(0)** moves the file pointer to the **beginning** of the file, and **raf.seek(raf.length())** moves it to the **end** of the file.

❖ Constructors of RandomAccessFile class

RandomAccessFile class defines the following constructors in Java. They are as follows:

1. RandomAccessFile(File fileObj, String mode):

2. RandomAccessFile(String filename, String mode): This constructor creates a random.

In both cases, the mode determines what kind of file access is permitted. For example:

- If the mode (in string) is **“r”**, the file can be **read-only**, but not written.
- If it is **“rw”**, the file is opened in **read-write** mode.

❖ RandomAccessFile Methods in Java

Methods	Description
1. void close():	This method closes the random access file stream and releases any system resources associated with the stream.
2. long getFilePointer():	This method returns the current offset (in bytes) from the beginning of the file to where the next read or write occurs.
3. long length():	This method returns the length (number of bytes) of this file.
4. int read():	It reads a byte of data from this file and returns -1 at the end of the stream.
5. int read(byte[] b):	It reads up to b.length bytes of data from this file into an array of bytes.
6. boolean readBoolean():	This method reads a boolean value from this file.
7. char readChar():	This method reads a character value from this file.
8. double readDouble():	The readDouble() method reads a double value from this file.
9. float readFloat():	The readFloat() method reads a float value from this file.
10. String readUTF():	The readUTF() method reads in a string from this file.
11. void seek(long pos):	This method sets the file-pointer (in bytes specified in pos), measured from the beginning of this file, at which the next read or write occurs.
12. void write(byte[] b):	This method writes b.length bytes from the specified byte array to this file, starting at the current file pointer.
13. void write(int b):	The write() method writes the specified byte to this file.
14. void writeBoolean(boolean v):	It writes a boolean value (one byte) to the file.
15. void writeChar(int v):	It writes a char to the file as a two-byte value, high byte first.
16. void writeDouble(double v):	It first converts the specified double value to a long value using the doubleToLongBits method in class Double, and then writes that long value to the file as an eight-byte quantity, high byte first.
17. void writeFloat(float v):	It first converts the specified float value to an int value using the floatToIntBits method in class Float and then writes that int value to the file as a four-byte quantity, high byte first.
18. void writeInt(int v):	It writes an int value to the file as four bytes, high byte first.
19. void writeLong(long v):	It writes a long value to the file as eight bytes, high byte first.
20. void writeUTF(String str):	It writes a string value to the file using modified UTF-8 encoding in a machine-independent manner.

Ex:1

```
import java.io.*;
class F1
{
    public static void main(String[] args)throws Exception
    {
        RandomAccessFile raf=new RandomAccessFile("raf1.txt", "rw");
        String s ="Hello Programmers How are You";

        raf.write(s.getBytes());
        raf.seek(0);
        System.out.println("Current pointer position:"+raf.getFilePointer());
        //Current pointer position:0

        int i=raf.read();
        System.out.println("Character at 0 position:"+(char)i);
        //Character at 0 position:H

        System.out.println("Current pointer position:"+raf.getFilePointer());
        //Current pointer position:1

        i=raf.read();
        System.out.println("Character at next position::"+(char)i);
        //Character at next position::e

        raf.seek(6);
        System.out.println("Current pointer position:"+raf.getFilePointer());
        //Current pointer position:6

        i=raf.read();
        System.out.println("Character at 6th position::"+(char)i);
        //Character at 6th position::P
        raf.close();
    }
}
```

Ex:2

```
import java.io.*;
class F1
{
    public static void main(String[] args)throws Exception
    {
        RandomAccessFile raf=new RandomAccessFile("raf1.txt", "rw");
```

```

String s ="Hello Programmers How are You";

byte [] b=s.getBytes();
raf.write(b);

// moving file pointer
raf.seek(2);//set the pointer at offset 2

byte[] bytes = new byte[17];

// storing file data
System.out.println(raf.read(bytes));//17
//read 17 bytes starts after offset 2
raf.close();

// printing file data
String ss=new String(bytes);
System.out.println(ss);
System.out.println(new String(bytes));//llo Programmers H
}
}

```

Ex:3

```

import java.io.IOException;
import java.io.RandomAccessFile;
class RandomIO
{
    public static void main(String[] args) throws IOException
    {
        // Create a random access file object.
        RandomAccessFile file = new RandomAccessFile("myfile.txt", "rw");

        // Writing to the file.

        file.writeChar('S');
        file.writeInt(2222);
        file.writeDouble(222.22);

        file.seek(0); // Moving file pointer to the beginning.

        // Reading data from the file.
        System.out.println(file.readChar()); //2
        System.out.println(file.readInt()); //2222
        System.out.println(file.readDouble()); //222.22
    }
}

```

```

// Moving the file pointer to the second item.

file.seek(2);
System.out.println(file.readInt()); //2222

// Go to the end and append a boolean value to the file.

file.seek(file.length());
file.writeBoolean(true);

// Since pointer is at the end, beyond the 4th item, brings the file pointer to the 4th item in
the file.

// Moving the file pointer to the 4th item.

file.seek(4);
System.out.println(file.readBoolean()); // Reading 4th item.
//true
file.close(); // Closing stream.
}
}

```

Ex:4

```

import java.io.IOException;
import java.io.RandomAccessFile;
class RandomIO1
{
    public static void main(String[] args) throws IOException
    {
        // Create a random access file object.
        RandomAccessFile file = new RandomAccessFile("raf1.txt", "rw");

        // Clear the file to destroy the old data if exists.
        file.setLength(0); // Empty file.

        // Write new integer values to the file.
        for(int i = 0; i <= 10; i++)
        {
            file.writeInt(i);
        }
    }
}

```

```
// Print the current length of the file.
System.out.println("Current length of file: " +file.length());
// Current length of file: 44

// Get the first number from the file.
// Bring the file pointer to the beginning.
file.seek(0);
System.out.println("First number: " +file.readInt());
// First number: 0

// Get the second number from the file.
System.out.println("Second number: " +file.readInt());
// Second number: 1

// Retrieve the number at 9th position.
// Move the file pointer to 9th position.
file.seek(9 * 4);
System.out.println("Ninth number: " +file.readInt());
// Ninth number: 9

// Modify the tenth number.
file.writeInt(222);
file.seek(10 * 4); // Moving pointer to 10th position.
System.out.println("Tenth number: " +file.readInt());
// Tenth number: 222

// Append a new number at the end of the file.
file.seek(file.length());
file.writeInt(333);

// Print the new length of the file.
System.out.println("New length of file: " +file.length());
// New length of file: 48

// Retrieve the new number. Move the file pointer to new number.
file.seek(11 * 4);
System.out.println("New number: "+file.readInt());
// New number: 333
}
}
```

EX:5

```
import java.io.*;
class NewClass
{
    public static void main(String[] args) throws Exception
    {
        // Creating a new RandomAccessFile - "raf.txt"
        RandomAccessFile raf1= new RandomAccessFile("raf2.txt", "rw");

        // public final void writeUTF(String str)
        //Writes a string to the file
        raf1.writeUTF("The purpose of our life is to be happy");

        raf1.seek(0);
        System.out.println("Use of readUtf() method : " + raf1.readUTF());
        //Use of readUtf() method : The purpose of our life is to be happy
    }
}
```