Computer Programming

Dr. Deepak B Phatak
Dr. Supratik Chakraborty
Department of Computer Science and Engineering
IIT Bombay

Recap: handling data in files

Recap

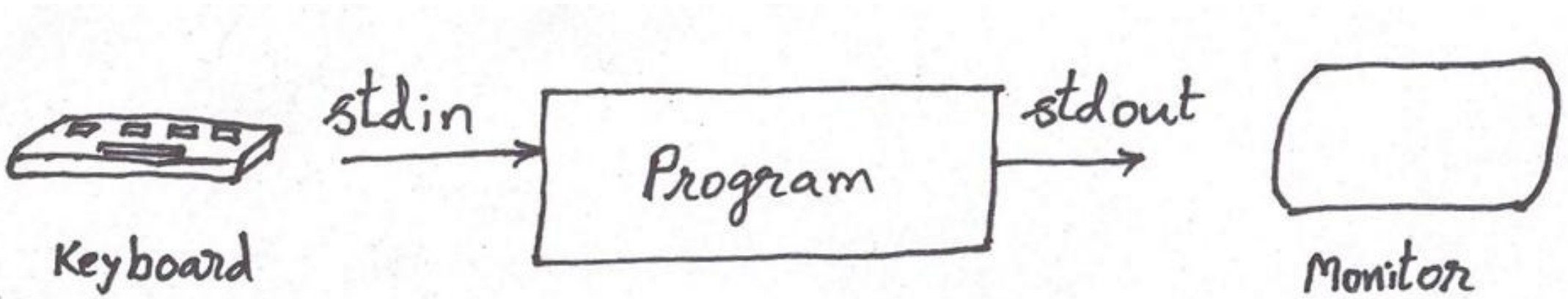
IIT Bombay

- Each file stored on an external device has
  - Name and a path (location), size, and access permissions
- All files are handled by the Operating System
- A C++ program can define and use FILE pointers
  - Name of the pointer is the internal filename for C++
  - It has to be “associated” with an external file

Normal I/O operations

IIT Bombay

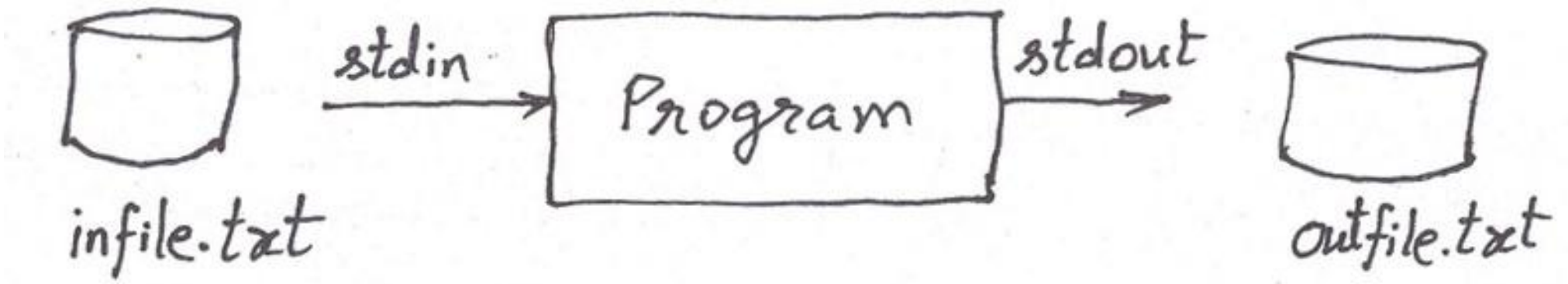
- stdin and stdout are automatically opened by OS
- These are automatically connected to keyboard and terminal



Redirection

IIT Bombay

$ myprog < infile.txt > outfile.txt



The printf() function

IIT Bombay

- Converts values as per a specified format string

```
int roll = 12345,  int batch =112;
printf("%5d %3d\n", roll, batch)
```

This will produce output line on stdout:

"12345 112\n"

scanf("format-string", &var, &var, ..)


```
int M, N; float x, y; char name[40];
scanf("%d %d %f %f %s", &M, &N, &x, &y, name);
```

- Any one of the following lines of text data will be interpreted correctly, with same values being assigned to variables

```
25   -78  .00763  345.29 Mynameischandra
  25    -78   7.63E-3  3.4529E2 Mynameischandra
```

Another example of scanf()

IIT Bombay

```
int a; float x; char itemcode[8];
// The input data line contains
//  123456fanbelt150.50
scanf("%6d%7s%f", &a, itemcode, &x);
printf("%6d\t%7s\t%6.2f\n", a, itemcode, x);
```

Different versions of these functions

- **Interpret from a string/create an output string**

  sprintf(s, "format string", expression, …)

  sscanf(s, "fomat-string", &var1, &var2, …)

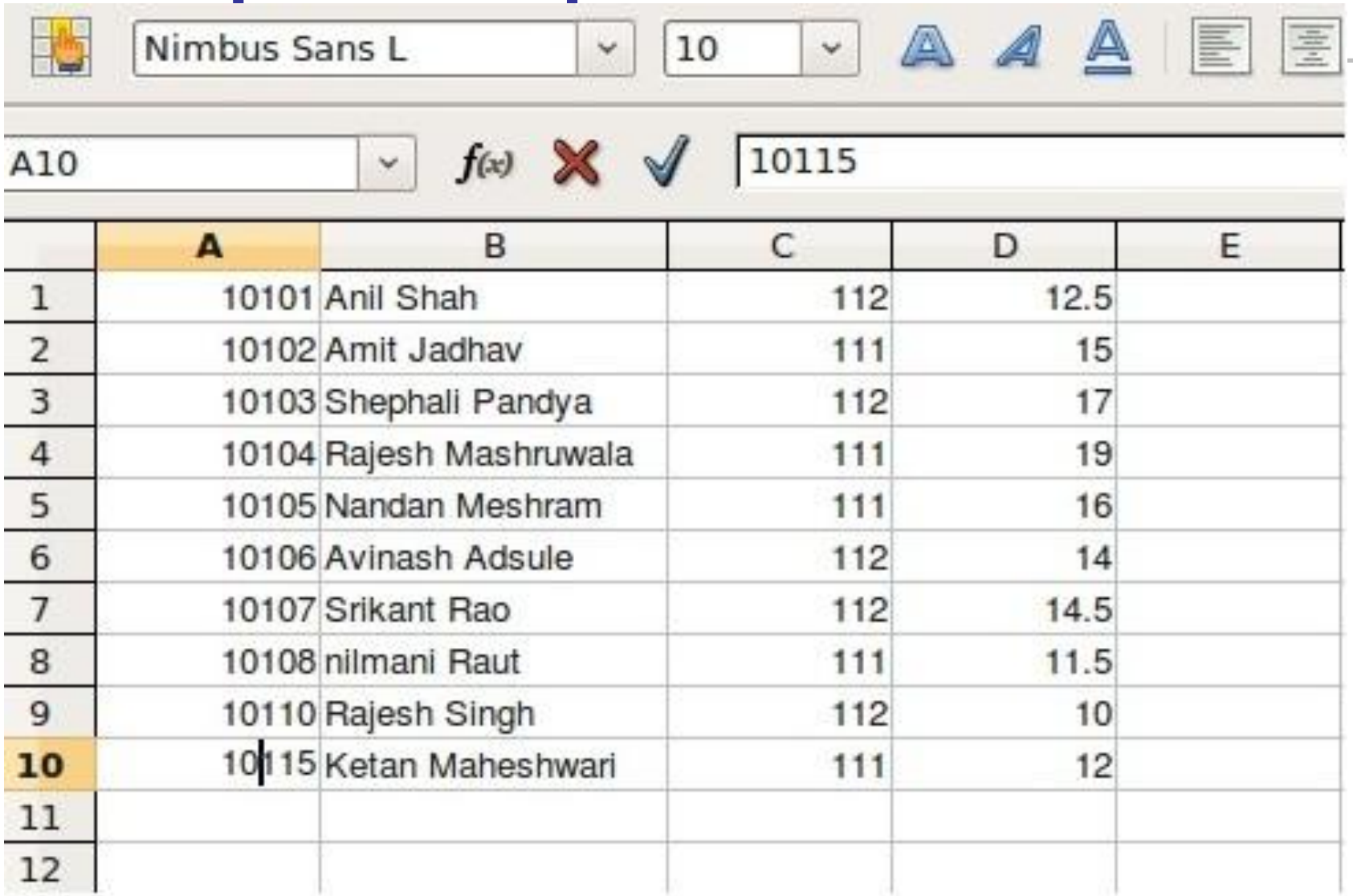
- **Interpret input from a linefrom a text file**

  fscanf(fpin, "fomat-string", &var1, &var2, …)

- **Output a formatted line to a text file**

  fprintf(fpout, "format string", expression, …)

Example of a spread-sheet



	A	B	C	D	E
1	10101	Anil Shah	112	12.5	
2	10102	Amit Jadhav	111	15	
3	10103	Shephali Pandya	112	17	
4	10104	Rajesh Mashruwala	111	19	
5	10105	Nandan Meshram	111	16	
6	10106	Avinash Adsule	112	14	
7	10107	Srikant Rao	112	14.5	
8	10108	nilmani Raut	111	11.5	
9	10110	Rajesh Singh	112	10	
10	10115	Ketan Maheshwari	111	12	
11					
12					

CSV data (Comma Separated Values)

10101,Anil Shah,112,12.5

…

10106,Avinash Adsule,112,14

10107,Srikant Rao,112,14.5

10108,Nilmani Raut,111,11.5

10110,Rajesh Singh,112,10

10115,Ketan Maheshwari,111,12

A program to process data from a CSV file

- Read one line from the input file in a string
  - `char linestr[80];`

For example, the first line is: 10101,Anil Shah,112,12.5

- Separate the four parts in four different strings
- Convert each part in internal form

```
int sr; char sn[30]; int sb; float sm;
```

… Process Data from CSV file

IIT Bombay

- Now put these four values, separated by blank spaces, together in a string

  outstr (char outstr[80];)

- write this string to the output file
- Repeat this procedure to process all lines from the input file

Program logic

IIT Bombay

Read one line of input file in linestr

While (not end-of-file for the input file){

Process input string, separate parts in four strings

convert each part and store in an appropriate variable

Prepare an output text string with these four values

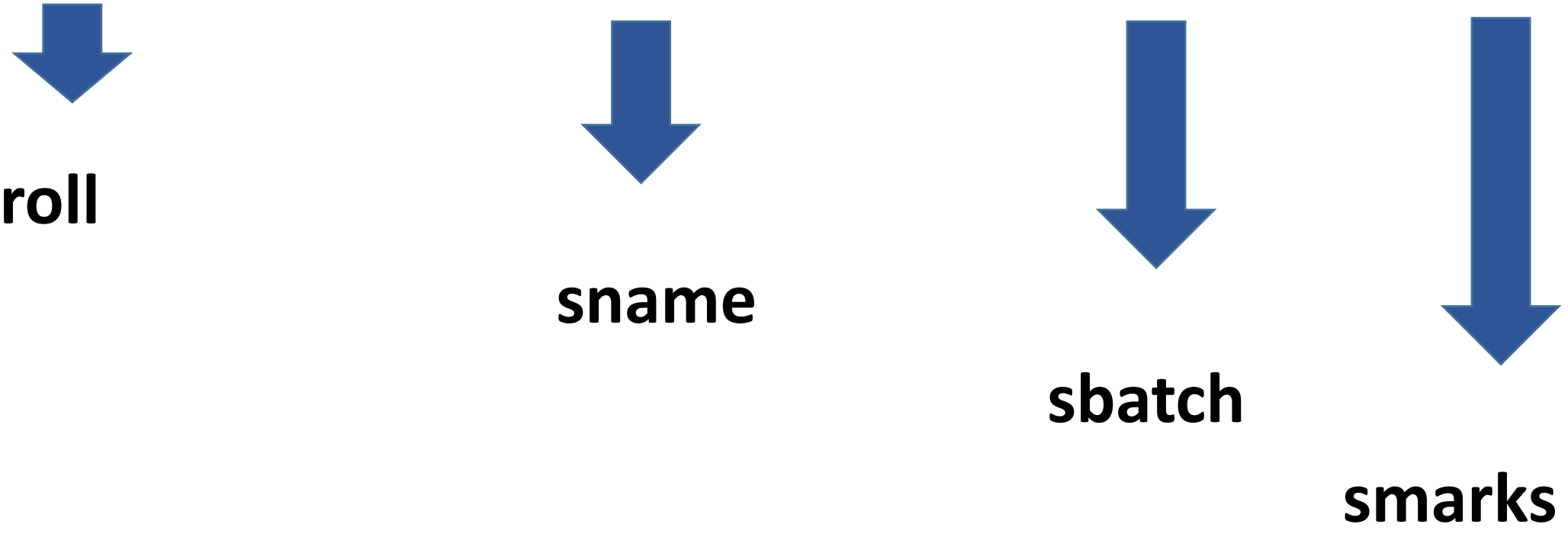
write this string (outstr) to output file

read next input line in linestr}

Separating parts in four strings

IIT Bombay

|1|0|1|0|1|,|A|n|I|I| |S|h|a|h|,|1|1|2|,|1|2|.|5|



Program …

IIT Bombay

```
#include <iostream>
#include <cstring>
#include <cstdio>
using namespace std;
int main() {
  char linestr[80]; char outstr[80];
  char sroll[6], sname[30], sbatch[4], smarks[10];
  int sr; char sn[30]; int sb; float sm;
  int i,j,k,N =0;
```

Program …

IIT Bombay

```
FILE *fpin; FILE* fpout;
fpin = fopen("CSV_data.txt", "r" );
if (fpin == NULL){
  cout << "Could not open file" << endl;
  return -1;
}
```

Program …

IIT Bombay

```
fpout = fopen ("marks_data.txt", "w");
if (fpout == NULL){
  cout << "Could not create output file" << endl;
  return -1;
}
```

Program …

IIT Bombay

```
/*Input  file is open at this point, read lines one by one */
  fgets(linestr, 79, fpin);
  while (!feof (fpin)){
     /* valid string, separate the parts */
     i =0; k =0;
```

Program …

IIT Bombay

```
while ((sroll[i++] = linestr[k++]) != ',');
sroll[i-1]='\0'; i=0;
while ((sname[i++] = linestr[k++])!= ',');
for (j = i-1; j<29; j++) sname[j] = ' ';
sname[29] ='\0'; i=0;
while ((sbatch[i++] = linestr[k++]) != ',');
sbatch[i-1] = '\0'; i=0;
while((smarks[i++]=linestr[k++]) != '\0');
```

Program …

IIT Bombay

```
/*extract relevent values from these strings */
sscanf (sroll, "%d", &sr);
sscanf(sname,"%s", sn);
sscanf(sbatch,"%d", &sb);
sscanf(smarks, "%f", &sm);
```

Program …

IIT Bombay

```
  sprintf(outstr, "%5d %30s %3d %5.2f\n",sr,sn,sb,sm);
  fputs(outstr, fpout);
  printf("%s", outstr);
  fgets(linestr, 79, fpin);
   N=N+1;
} //End of while loop, input file has been processed
```

Program …

IIT Bombay

```
  cout << "\ninput file has been read and printed\n";
  cout << "output file Marks_data.txt created\n";
  cout << N << "  records written to output file\n";
  fclose(fpin);  fclose(fpout);
  return 0;
}
```

A record structure

IIT Bombay

```
struct studentinfo {
    int roll;
    char name[30];
    int batch;
    float marks;
}
```

Size and elements of a structure

- We define a structure variable s

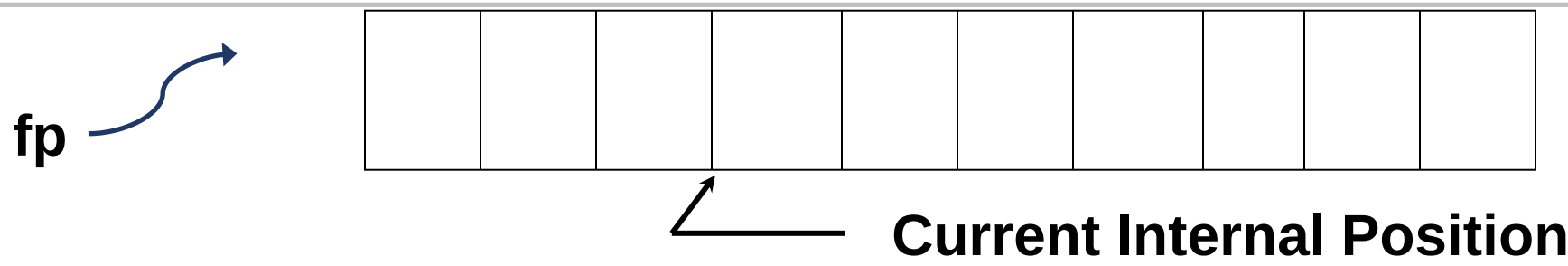
  struct studentinfo s;
- Individual elements of s can now be accessed by

  s.roll, s.name, s.batch , and s.marks
- The size (in bytes) of a structure can be found by

  int rec_size;

  rec_size = sizeof(struct studentinfo)

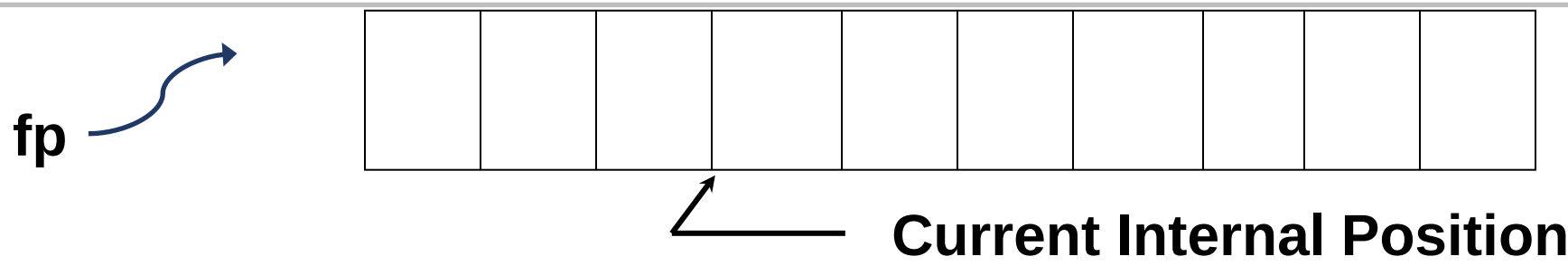
Accessing data in a binary file on disk



- Upon opening the file, position is set at 0 (beginning)
- Reading/writing happens at this position
- Each I/O operation advances internal position, by the number of bytes read or written

Accessing records in a binary file on disk

IIT Bombay

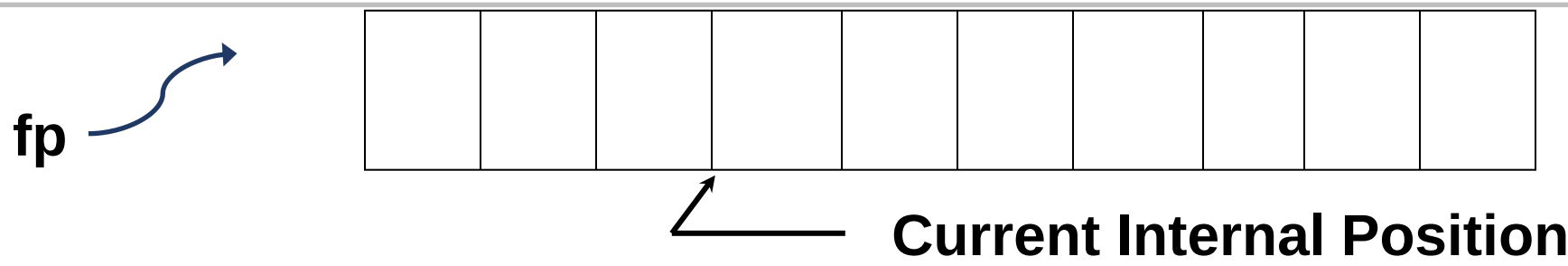


- Current internal position can be found using a function in C++, named ftell()

```
long POS;    POS = ftell(fp);
```

Accessing records in a binary file on disk

IIT Bombay

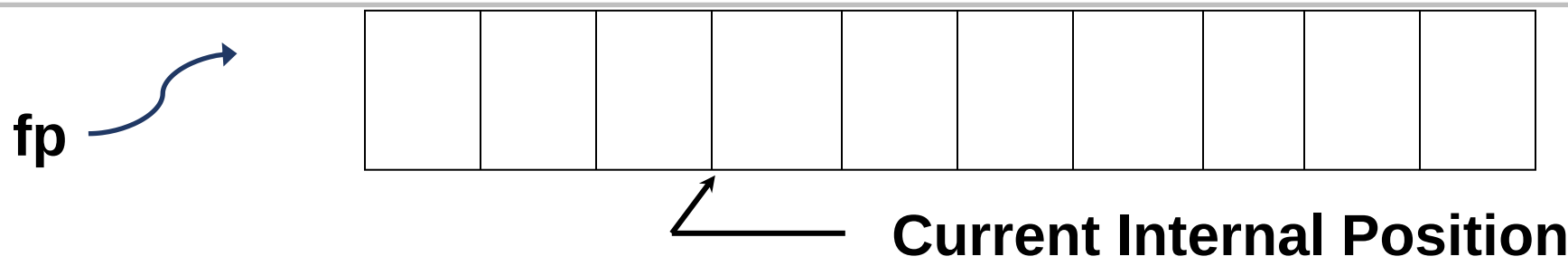


- One can set the position to any desired point, say P bytes (P should be declared as type long)

  fseek (fp, P, SEEK_SET); //Count from beginning

Accessing records in a binary file on disk

IIT Bombay



- Current internal position can be found using a function in C++, named ftell()

  long POS; POS = ftell(fp);

- One can set the position to any desired point, say P bytes (P should be long)

  fseek (fp, P, SEEK_SET); //Count from begining

Fixed length records

IIT Bombay

- We know the record size (number of bytes in a record), say S
- If we know the relative position r, of the record of a student in the file, then we can directly read the data for that student
  - S* (r-1) will be the starting byte position of the record
  - Next S bytes will contain the record

Some relevant C++ functions

- For reading a record in struct variable S

  ```
  fread(&s, rec_size, 1, fp);
  // Note use of pointer to struct
  ```

- For writing one record from a struct variable

  ```
  fwrite(&s, rec_size, 1, fp);
  ```

- To reset internal pointer to beginning of file

  ```
  Rewind(fp);
  ```

Program …

IIT Bombay

```
fp_input = fopen("markdata.txt", "r" );
fp_output=fopen("studentdb","wb");
if (fp_output == NULL){
    cout << "Could not open output file"<< endl;
    return -1;
}
```

Program …

IIT Bombay

```
fscanf (fp_input, "%d %s %d %f",&r,n,&b,&m);
while (!feof (fp_input)) {
   count++; s.roll=r, s.batch=b;
   s.marks=m; strcpy(s.name, n);
   fwrite(&s, rec_size, 1, fp_output);
   fscanf (fp_input, "%d %s %d %f",&r,n,&b,&m);
}
```

Program …

IIT Bombay

```
  cout << "marks data file read and printed\n";
  cout << "Database created for student info\n";
  cout << "Total records written:  "<<count<<endl;
  fclose(fp_input);
  fclose(fp_output);
  return 0;
}
```

Program to process studentdb

IIT Bombay

```
#include <iostream>
#include <cstdio>
using namespace std;
 struct studentinfo{
    int roll;
    char name[30];
    int batch;
    float marks;
};
```

Program …

IIT Bombay

```
int main() {
  struct studentinfo s;
  int r, recnum, rec_size, found =0; long POS;
  rec_size = sizeof(struct studentinfo);
  cout << "Size of each record is:  " <<rec_size<<endl;
```

Program …

IIT Bombay

```
FILE *fp;
fp = fopen("studentdb", "rb+" );
if (fp == NULL){
   cout << "Could not open database file" << endl;
   return -1;
}
int count=0; found=0;
```

Program …

IIT Bombay

```
rewind(fp);
cout << "searching for marks of roll 10105";
r = 10105; found =0;
POS = ftell(fp);
// find marks for roll number 10105
```

Program …

IIT Bombay

```
do {
    fread(&s,rec_size,1, fp);
    if (s.roll ==r){
       // found student, print record
     cout << endl << "Found roll "<< r;
     found = POS/rec_size +1;
     cout << " at record no. " << found <<"\nThe name and marks are: ";
     printf("%s %5.2f \n", s.name,s.marks);
     break;
  }
```

Program …

IIT Bombay

```
} while (!feof (fp));
  if (found == 0){
     cout << "\nroll number not found in database\n";
  }
   cout << " ------------"<<endl;
```

Program …

IIT Bombay

```
cout << "demonstrating direct access to records" << endl;
cout << "read and display 6th record"<< endl<< endl;
// find and print 6th record in database file
recnum = 6;
POS = (recnum-1) * rec_size;
```

Program …

IIT Bombay

```
fseek (fp, POS, SEEK_SET);// set the file position
fread(&s, rec_size, 1, fp);
cout << "Record starting at byte position " << POS << " is\n";
printf("%5d %30s %3d %5.2f \n", s.roll, s.name,s.batch,
        s.marks);
cout << "Record Number is: " << recnum << endl;
```

Program …

IIT Bombay

```
// update Nilamani's marks to 93.5
// His roll number is 10108.
// We access 8th record starting at 7 * rec_size)
 cout << " ------------"<<endl;
 cout << "read and update Nilamani Raut's marks" << endl;
 cout << "Nilamani's roll number is 10108"<< endl;
 r = 10108;
```

Program …

IIT Bombay

```
recnum = r-10100;
POS = (recnum-1)*rec_size;
fseek (fp, POS, SEEK_SET);
fread(&s,rec_size, 1,fp);
cout << "Record for Nilamani is\n";
printf("%5d %30s %3d %5.2f \n", s.roll, s.name,s.batch,
        s.marks);
```

Program …


```
s.marks = 93.5;
// previous read has advanced the internal
// position indicator, set it back correctly
fseek (fp, POS, SEEK_SET);
fwrite(&s, rec_size,1, fp);
```

Program ,,,

IIT Bombay

```
// verify correct data is written
fseek (fp, POS, SEEK_SET);
fread(&s, rec_size, 1,fp);
cout << "--------Updated record in database file is\n";
printf("%5d %30s %3d %5.2f \n",
            s.roll, s.name,s.batch, s.marks);
```

Program …

IIT Bombay

```
  fclose(fp);
  return 0;
}
```

Practice Problem

- Assume that the binary file 'Mobile.bin' contains mobile information like 'Mobile Number' and 'Mobile recharge amount'.

a) First, write a main program to appropriately open a binary file for reading and writing. The main program should make a call to the two functions 'mobileNumberExists' and 'updateRechargeAmount'.

The function 'updateRechargeAmount' should be called only if the function 'mobileNumberExists' returns true. Declare the two functions with the required rerutn value and the paramenters that need to be passed.

b) Write the first function 'mobileNumberExists'to check whether the mobile entered by the user exists in the binary file 'Mobile.bin'. If it exists, the function should return true to the main program(from where it was called).

c) Write the second function 'updateRechargeAmount' to update the binary file with the new recharge value for that mobile number.