EXAMPLES of BINARY file operationsusing the library <stdio.h>

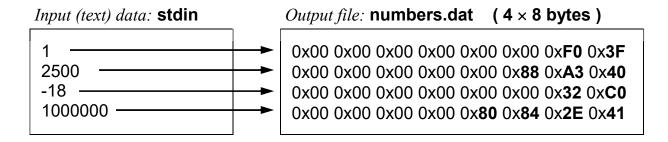
BINARY files composed of sequence of numbers:

```
BIN-1: Store the sequence of numbers (from keyboard) into the file ( fwrite )
BIN-2: Read and display the content of a binary file ( fread )
BIN-3: Read the portion from inside the binary file ( fseek, fread )
BIN-4: Append some numbers at the end of the binary file ( a - append )
BIN-5: Update / overwrite the portion of binary file ( fseek, ftell, fwrite )
BIN-6: Filtering – copying numbers matching the specified criteria using a loop:
while( fread(...)==1 )
```

if(...)

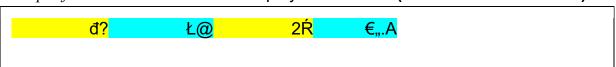
fwrite(...);

Example BIN-1: Writing a sequence of numbers to a binary file



The contents of the "number.dat" file viewed with a text editor (for example, with the system's "Notepad") looks as follows:

Output file: numbers.dat - displayed as a text ($32 = 4 \times 8$ characters)



```
|| The program iteratively reads consecutive numbers from the keyboard into the buffer
Il and immediately copies the contents of the variable buffer (8 bytes) to the binary file
#include <stdio.h>
int main()
{ FILE* my_file = fopen( "numbers.dat", "wb" );
  if( my_file )
     {
        Il retrieving the number of read-in data (in the example above it is equal 4)
        int how many numbers;
        printf("Enter how many numbers you want to save in the file: ");
        scanf("%d", &how many numbers);
        double buffer;
        for(int i=0; i< how_many_numbers; i++)</pre>
          {
             printf("\nEnter the next number: ");
             scanf("%lf",&buffer);
             fwrite(&buffer, sizeof(double), 1, my file );
        fclose(my file);
  return 0;
}
```

Binary write: fwrite(data_pointer, block_size, amount_of_blocks, file)

Example BIN-2: Reads and displays the entire contents of a binary file

```
If The program iteratively gets (loads) from the file "numbers.dat",
Il subsequent 8-byte portions of data, to the auxiliary variable buffer of type double
Il and then displays them on the screen (send them to the text file stdout)
#include <stdio.h>
int main()
   FILE* my file = fopen( "numbers.dat", "rb" );
   if(!my file)
     {
        printf("File open error ");
        getchar();
   else
     {
        double buffer:
        int counter=0;
        Il iterative loop to get successive portions of sizeof (double)
        while( fread(&buffer, sizeof(double), 1, my_file)==1 )
           {
              Il displaying the loaded numbers on the console screen
              printf("%f\n", buffer );
              | Here, further processing of the loaded number can take place . . .
              If for example, summation:
                                             sum = sum+buffer;
              Il or searching the minimum:
                   if( buffer<minimum) minimum=buffer;</pre>
              || In this example, it is just the count of the number of loaded numbers:
              counter ++;
        fclose(my file);
        printf("\n End of reading. Number of loaded data = %d",counter);
  return 0;
}
```

Binary read: **fread**(**data_pointer**, **block_size**, **amount_of_blocks**, **file**)
The **fread** and **fwrite** functions return the number of correctly processed blocks, this return value may be used to control read / write errors

Example BIN-3: Read the pointed number from inside of a binary file

```
|| The program moves the file pointer ("read head")
Il to the indicated place (distance in bytes from the beginning of the file)
|| and then takes 8-bytes into a variable buffer of type double
Il and displays its content on the console screen
#include <stdio.h>
int main()
{ FILE* my_file = fopen( "numbers.dat", "rb" );
   if( !my_file )
     printf("File open error ");
  else
     { long position_of_item = 30; // \leftarrow can by any position inside the file}
        Il file pointer shift command
        fseek(my_file, position_of_item * sizeof(double), SEEK_SET);
        double buffer for number;
        Il reading 8-bytes, from the indicated file location, into the buffer for number
        if( fread(&buffer for number, sizeof(double), 1, my file)==1)
           printf("Loaded number = %f\n",buffer_for_number);
        else
           printf("Some error while reading the number from the file ");
        fclose(my file);
   printf("\nEnd of reading. Press ENTER ");
  fflush(stdin); getchar();
   return 0;
}
```

Example BIN-4: Append three numbers (22,44,66) to the end of the file

```
#include <stdio.h>
int main() {
 if(!my file)
   printf("File open error ");
  else
   { double number;
     for(int i=1; i<=3; i++)
         number = i*22:
         fwrite(&number, sizeof(double), 1, my file);
     fclose(my file);
 return 0;
}
```

Example BIN-5: Replace/overwrite the value of a number in a binary file

```
Il The program moves the file pointer to the specified position, in relation to the beginning
If and then (over)writes there 8-bytes from the variable new value of type double
#include <stdio.h>
int main()
{ FILE* my_file = fopen( "numbers.dat", "r+b" ); ← "+" means ,, modification"
  if(my file)
     Il reading the file size (and calculating the number of all double numbers in it )
     fseek(my file,0,SEEK END);
     long file length in bytes = ftell(my file);
     long all numbers = file length in bytes / sizeof(double);
     printf("This file contains %ld numbers <double>\n\n", all numbers);
     Il determining (loading) the position of the modified number
     long overwritten_position;
     printf("Enter the position of the number to be overwritten: ");
     scanf("%|d", &overwritten position);
     Il checking if the given item is correct (is it inside the file?)
     if(overwritten position >= 0 && overwritten position < all numbers)
        { double new value;
          printf("\nEnter a new value for the number: ");
          scanf("%|f",&new value);
          Il moving the file pointer and writing the new value to the file
          fseek(my file, overwritten position*sizeof(double), SEEK SET);
          fwrite(&newvalue, sizeof(double), 1, my file);
     fclose(plik);
  return 0;
}
```

Example BIN-6: Filtering - copying numbers according to a given criterion

```
#include <stdio.h> // copying only positive numbers to the second binary file
int main()
{ double number;
  FILE* input_file = fopen( "numbers.dat", "rb" );
  FILE* file_positive_only = fopen("only positive numbers.dat", "wb");
  if( input file!=NULL && file positive only!=NULL )
     while( fread(&number, sizeof(double), 1, input file)==1 )
       if( number>0)
          fwrite(&number, sizeof(double), 1, file positive only);
  fclose(input file);
  fclose(file positive only);
  return 0;
```