DATOTEKE

Dragan dr Dinu

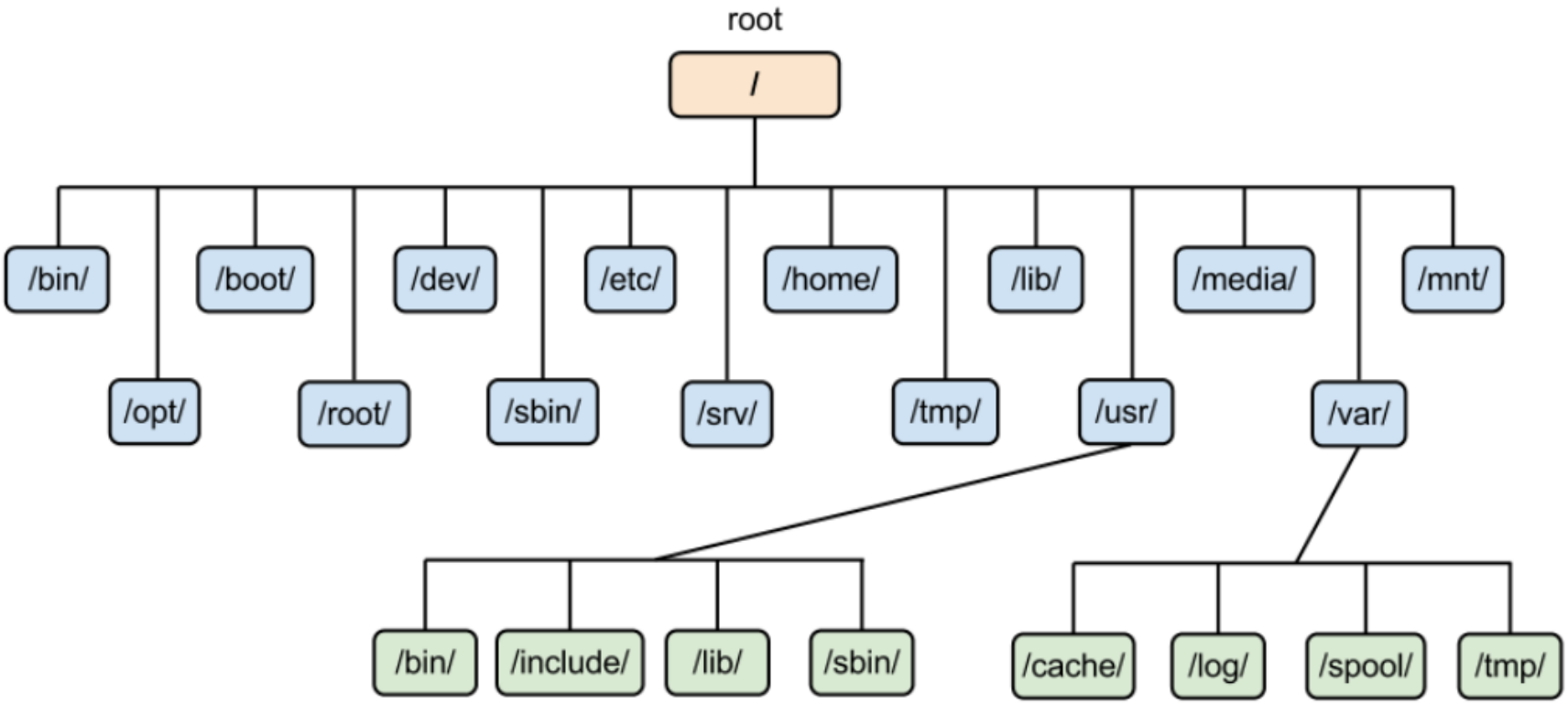
OSNOVNO O DATOTEKAMA

- Za trajno skladištenje podataka
- Strukture podataka na masovnoj memoriji
- Prema načinu pristupa podacima u datoteci dele se na:
  - **sekvencijalne datoteke**
  - **direktne datoteke**
- Prema organizovanju podataka u datoteci dele se na:
  - **sekvencijalne datoteke**
  - **direktne (hash) datoteke**
  - **indeks-sekvencijalne datoteke**
- Prema prirodi podataka u detoteci dele se na:
  - **tekstualne datoteke**
  - **binarne datoteke**

SISTEMI ZA RAD SA DATOTEKAMA

- Sistem za rad sa datotekama (engl. filesystem) kontroliše kako se čuvaju podaci i kako im se pristupa
- Globalno definiše način na koji računar organizuje, imenuje, čuva i manipuliše datotekama
- Bez sistema za rad sa datotekama, podaci zapisani na memorijskom medijumu predstavljali bi jedinstvenu celinu, tj. ne bi mogli da kažemo gde se neka informacija završava, a sledeća počinje
- Primeri sistema za rad sa datotekama:
  - FAT (FAT16, FAT32), NTFS, ext (2, 3, 4), UDF, …

PRIMER – LINUX FILESYSTEM (EXT)



Izvor:https://freedompenguin.com/articles/how-to/learning-the-linux-file-system/

TOK RADA SA DATOTEKOM U C JEZIKU ...

1. **Deklarisanje datotečne promenljive**

   FILE *dat_prom;

2. **Otvaranje datoteke**

   FILE *fopen(const char *naziv_datoteke, const char *rezim);

Režimi rada	
“r” / **“rt”** **“rb”**	Otvori za čitanje **tekstualnu** ili **binarnu** datoteku, počinje od početaka datoteke. Ako datoteke ne postoji, vratiće NULL.
“w”/ **“wt”** **“wb”**	Otvori za pisanje **tekstualnu** ili **binarnu** datoteku, počinje od početka datoteke. Gubi se stari zapis. Ako datoteka ne postoji, kreiraće novu.
“a”/ **“at”** **“ab”**	Otvori za dodavanje **tekstualnu** ili **binarnu** datoteku, počinje od kraja datoteke i omogućuje dodavanje novih zapisa. Čuva se stari zapis. Ako datoteka ne postoji, kreiraće novu.
“r+” / **“rt+”** ili **“rb+”** **“w+”**/ **“wt+”** ili **“wb+”** **“a+”**/ **“at+”** ili **“ab+”**	Čitanje i pisanje od početka. Gubi se stari zapis. Čitanje i pisanje od početka. Gubi se stari zapis. Čitanje i pisanje od kraja. Čuva stari zapis.

... TOK RADA SA DATOTEKOM U C JEZIKU

3. Čitanje ili pisanje podataka u datoteku

- koriste se različite funkcije u zavisnosti od vrste datoteke
- čitanje do kraja pomeranjem internog pokazivača datoteke:

```
int *feof(FILE *dat_prom);
```

4. Zatvaranje datoteke

```
int fclose(FILE *dat_prom)
```

STRUKTURA FILE

- Definisana u zaglavlju stdio.h
- Direktorijum (folder) je samo specijalna vrsta fajla – fajl fajlova
- Pruža neophodne informacije o datoteci ili toku koji obavlja
- ulazne i/ili izlazne operacije, primer iz K&R:

```
typedef struct {
        short level;
        short token;
        short bsize;
        char fd;
        unsigned flags;
        unsigned char hold;
        unsigned char *buffer;
        unsigned char *curp;
        unsigned istemp;
} FILE;
```

Deskriptor datoteke sadrži atribute datoteke: naziv, veličina, redni brojevi blokova, vreme nastanka, izmene, pristupa, ...

TEKSTUALNE DATOTEKE

TEKSTUALNE DATOTEKE ...

- Sadržaj se interpretira kao ASCII (UNICODE), čak i kontrolni karakteri

- Funkcije za prenos znakova sa konverzijom:

int fscanf(FILE *dat_prom, const char *format [,adresa, ...])

int fprintf(FILE *dat_prom, const char *format [,prom, ...])

  - Isto kao i odgovarajuće funkcije za rad sa **stdin** i **stdout**
  - Navodi se datotečna promenljiva kako bi se znalo odakle se čita, tj. gde se piše

... TEKSTUALNE DATOTEKE

- Funkcije za prenos karaktera (bez konverzije):

Funkcije:	
int fgetc(FILE *dat_prom)	Funkcija koja čita iz tekstualne datoteke jedan karakter a koji vraća svojim identifikatorom
int getc(FILE *fajl_prom)	Makro koji čita iz tekstualne datoteke jedan karakter a koji vraća svojim identifikatorom
int fputc(int znak, FILE *fajl_prom)	Funkcija koja upisuje znak u tekstualnu datoteku, dok svojim identifikatorom vraća taj isti znak ili kod greške
int putc(int znak, FILE *fajl_prom)	Makro koji upisuje znak u tekstualnu datoteku, dok svojim identifikatorom vraća taj isti znak ili kod greške
char *fgets(char *str, int n, FILE *stream)	Funkcija koja čita iz tekstualne datoteke niz od n-1 znakova ili dok ne naiđe na znak '\0'.
int fputs(const char *str, FILE *fajl_prom)	Funkcija koja upisuje u tekstualnu datoteku niz znakova sa '\0' terminatorom

TEKSTUALNE DATOTEKE – PRIMERI

- **ZADATAK1:**
  Napisati program koji isčitava tekst iz tekstualnog fajla i isti ispisuje na ekran. Pretpostaviti da u jednom redu tekstualnog fajla može biti maksimalno 255 karaktera.

- **VEŽBA1:**
  Proširiti program tako omogući korisniku da unese naziv tekstualnog fajla.

- **VEŽBA2:**
  Napisati program koji omogućuje korisniku da unosi tekst sa tastature koji će se čuvati u tekstualni fajl po korisnikovom izboru.

- **VEŽBA3:**
  Napisati program koji kopira sadržaj jednog tekstualnog fajla u drugi po korisnikovom izboru.

TEKSTUALNE DATOTEKE – ZADATAK1


```
#include <stdio.h>
#include <stdlib.h>

#define MAKSKARAKT 255

int main()
{
    FILE *dt;

    char nazivDat[31] = "poruka.txt";
    char red[256];

    if ((dt = fopen(nazivDat,"r")) == NULL)     // Otvaranje datoteke sa proverom prava na citanje(r)
    {
        printf("\nGreska prilikom otvaranja datoteke \'%s\' za citanje.\n", nazivDat);
        exit(EXIT_FAILURE);                     // Prevremeni izlaz iz programa
    }

    while (fgets(red, 255, dt) != NULL)         // Citanje stringova maksimalne duzine 255 iz ulazne datoteke
        printf("%s", red);

    printf("\n");

    fclose(dt);                                 // Zatvaranje datoteke

    return 0;
}
```

BINARNE DATOTEKE

Dragan dr Dinu

BINARNE DATOTEKE ...

- Sadržaj se interpretira kao n-torka bitova (najčešće celobrojni umnožak bajta)
- Može se tumačiti da je organizovana kao **struct**
- Sadržaj binarne datoteke čita se pomoću funkcije:

```
int fread(void *gde, int vel_blok, int br_blok, FILE *dat_prom)
```

- Čita od trenutne pozicije internog pokazivača datoteke označenog sa **dat_prom**
- Čita se **br_blok** blokova, gde je svaki blok veličine **vel_blok**
- Pročitane vrednosti smeštaju se u memoriju počev od adrese **gde**

... BINARNE DATOTEKE

- U binarnu datoteku se piše pomoću funkcije:

  int fwrite(void *gde, int vel_blok, int br_blok, FILE *dat_prom)

- Piše se od trenutne pozicije internog pokazivača datoteke označenog sa **dat_prom**

- Upisuje se **br_blok** blokova, gde je svaki blok veličine **vel_blok**

- Vrednosti koje treba upisati u datoteku, čitaju se iz memorije počev od adrese **gde**

POZICIONIRANJE UNUTAR DATOTEKE

- Moguće je manipulisati internim pokazivačem datoteke
- Pozicija internog pokazivača saznaje se upotrebom funkcije:

  long ftell(FILE *dat_prom)

- Na početak datoteke se interni pokazivač pomera pomoću funkcije:

  void rewind(FILE *dat_prom)

- Funkcija za pomeranje internog pokazivača datoteke:

  int fseek(FILE *dat_prom, long ofset, int reper)

Moguće vrednosti parametra *reper*	
SEEK_SET	***ofset*** se računa u odnosu na početak fajla.
SEEK_CUR	***ofset*** se računa u odnosu na trenutnu poziciju internog pokazivača.
SEEK_END	***ofset*** se računa u odnosu na kraj fajla.

BINARNE DATOTEKE – PRIMERI

- **ZADATAK2:**
  Napisati program koji upisuje elemente niza od 10 prirodnih brojeva u binarnu datoteku.

- **ZADATAK3:**
  Napisati program koji čita elemente niza od 10 prirodnih brojeva iz binarne datoteke (sačuvane u prethodnom primeru).

- **VEŽBA4:**
  Napisati program koji vodi evidenciju o polaznicima kursa. Maksimalno ima 40 polaznika. Svaki polaznik opisan je brojem lične karte (koji ga jedinstveno identifikuje), imenom, prezimenom i nizom u kojem se čuva informacija o kursevima koje polaže. Prilikom gašenja programa, podaci se memorišu u binarnu datoteku. Prilikom pokretanja programa podaci se isčitavaju iz binarne datoteke u niz. Omogućiti korisniku da unosi, briše, modifikuje podatke o polaznicima kursa, kao i prikaz podataka o svim polaznicima.

BINARNE DATOTEKE – ZADATAK2


```
#include <stdio.h>
#include <stdlib.h>

#define VELICINA 10

int main()
{
  int i, niz[VELICINA];
  FILE *dp;

  for (i = 0; i < VELICINA; i++)                    // Inicijalizacija niza niz[]
    niz[i] = i + 1;

  if ( (dp = fopen("podaci.dat", "wb")) == NULL)    // Otvaranje datoteke u binarnom modu
  {
    fprintf(stderr, "Greska pri otvaranju datoteke.");
    exit(1);
  }

  if (fwrite(niz, sizeof(int), VELICINA, dp) != VELICINA)     // Ispis niz[] u datoteku
  {
    fprintf(stderr, "Greska pri ispisu u datoteku.");
    exit(1);
  }

  fclose(dp);

  return 0;
}
```

BINARNE DATOTEKE – ZADATAK3


```
#include <stdio.h>
#include <stdlib.h>

#define VELICINA 10

int main()
{
  int i, niz[VELICINA];

  FILE *dp;

  if ( (dp = fopen("podaci.dat", "rb")) == NULL)          // Otvaranje datoteke za citanje u binarnom modu
  {
    fprintf(stderr, "Greska pri otvaranju datoteke.");
    exit(1);
  }

  if (fread(niz, sizeof(int), VELICINA, dp) != VELICINA)  // Unos podataka u niz niz[]
  {
    fprintf(stderr, "Greska pri citanju datoteke.");
    exit(1);
  }

  fclose(dp);

  puts("Brojevi u datoteci PODACI.DAT su:");

  for (i = 0; i < VELICINA; i++)                          // Ispis podataka na ekran
    printf("\t%d\n", niz[i]);

  return 0;
}
```

BINARNE DATOTEKE – ZADATAK4 …

- **ZADATAK4:**
  Napisati program za praćenje proizvodnje u 8 fabrika. Svaka fabrika radi u 4 smene. Prati se ukupan broj proizvoda za svaku fabriku u svakoj smeni. Podaci o proizvodnji zapisuju se u binarnu datoteku proizvodnja.dat u obliku binarne reci sledeceg formata:

Smena	Fabrika	Proizvodi
2 bita	4 bita	10 bita

Program treba da omogući čitanje podataka iz datoteke i prikaz rezultata po smenama za izabranu fabriku.

… BINARNE DATOTEKE – ZADATAK4 …


```
#include <stdio.h>
#include <conio.h>
#include <string.h>

#define SMENA 4
#define FABRIKA 8

typedef unsigned short TProizvodnja[ SMENA ][ FABRIKA ];

// Prototipovi funkcija
char OsnMeni( );                    // Prikaz osnovnog menija
void Ispis (TProizvodnja Isp);      // Ispis podataka
```

… BINARNE DATOTEKE – ZADATAK4 …


```
/* main funkcija */
int main() {
   FILE *dat;              // datotecna promenljiva
   char izvor[30],         // naziv datoteke
   kraj = 0;               // pomocna promenljiva
   TProizvodnja proizvodnja;      // matrica 4 smene x 8 fabrika

   unsigned short znak;                    // pomoćna promenljiva za učitavanje binarnih reči
   unsigned short      maskaS =0xc000,     // maska za ekstrakciju smene, C = 1100
                       maskaF =0x3800,     // maska za ekstrakciju fabrike, 38 = 0011 1000
                       maskaP =0x07FF;     // maska za ekstrakciju prometa, 7FF = 0111 1111 1111

   printf("\nUnesite naziv izvorne datoteke: ");
   scanf("%s", izvor);

   if ((dat = fopen(izvor, "rb")) == NULL)       // otvaranje datoteke za binarno citanje (rb)
      printf("\n\n\n GRESKA: datoteka \'%s\' se ne moze otvoriti!\n\n", izvor);
```

… BINARNE DATOTEKE – ZADATAK4 …


```
    else {
      while (!feof(dat)) {              // feof funkcija vraca != 0 ako je stigla do kraja datoteke
        /* citanje jednog unsigned short int iz datoteke i njegovo smestanje u promenljivu 'znak' */
        fread(&znak, sizeof(short), 1, dat);
        /* pomocu maskaS i maskaF odredjuje se pozicija el. u matrici, a sa maskaP odredjuje se
        proizvodnja */
        proizvodnja[(znak & maskaS) >> 14][(znak & maskaF) >> 11] = znak & maskaP;
      }
      fclose(dat); // zatvoranje datoteke
      do {
        switch (OsnMeni()) {
          case '1': Ispis(proizvodnja); break;
          case '2': IspisSmena(proizvodnja); break;
          case '0': kraj = 1;
        }
      } while (!kraj);
    }
}
```

… BINARNE DATOTEKE – ZADATAK4 …


```
/* funkcija ispisuje meni i vraca izabranu stavku menija */
char OsnMeni() {
   char odg;

   printf("\n\n\n\nProgram za prikazivanje podataka iz datoteke ");
   printf("\n----------------------------------------");
   printf("\n        1. Ispis");
   printf("\n        2. Ispis po smenama");
   printf("\n        0. Kraj rada.");
   printf("\n----------------------------------------");
   printf("\nVas Izbor je (0-2) --> ");

   do {
      odg = (char)getch();
      if (odg <'0' || odg >'2')
         printf("\b \b\a");
   } while (odg < '0' || odg > '2');
   return (odg);
}
```

… BINARNE DATOTEKE – ZADATAK4 …


```
/* Prikaz podataka iz matrice */
void Ispis(TProizvodnja proizvodnja) {
   int i, j;

   printf("\n\n\ns\\p 1. 2. 3. 4. 5. 6. 7. 8.");

   for (i = 0; i < SMENA; ++i) {
      printf("\n %d.", i+1);

      for (j = 0; j < FABRIKA; ++j) {
         printf(" %-6hu", proizvodnja[ i ][ j ]);
      }
   }

   getch();
}
```

… BINARNE DATOTEKE – ZADATAK4


```
/* Prikaz ukupne kolicine proizvoda za izabranu smenu */
void PoProd(TProizvodnja proizvodnja) {
   int i, j;

   printf("\n\nUnesite smenu po kojoj se pretrazuje proizvodnja fabrika: ");

   i = -1;     // i inicijalizujemo na nedozvoljenu vrednost jer scanf ne mora uspeti

   do {
      fflush(stdin);
      scanf("%d",&i);

      if (i < 1 || i > 4)
         printf("GRESKA: Redni broj smene mora biti u intervalu [1..4]\n");
   } while(i < 1 || i > 4);

   printf("Proizvodnja u %d. smeni je:\n", i);

   for (j = 0; j < FABRIKA; ++j)
      printf("\t %d", proizvodnja[ i-1 ][ j ]);

   getch();
}
```