JMBAG

IME I PREZIME

Programiranje 2 – kolokvij, 2. 5. 2025.

Napomene. Svako rješenje napišite isključivo na papir sa zadatkom jer jedino njega predajete. Pomoćne račune smijete raditi na drugim papirima koje će vam dati dežurni asistent. U svim zadacima **obavezno** pišite postupak!
Dozvoljeno je korištenje isključivo pribora za pisanje i brisanje, te službenog podsjetnika. Kalkulatori, te razne tablice, papiri i sl. nisu dozvoljeni! **Mobitele** isključite i pospremite daleko od sebe! Ako se ustanovi da **kod sebe** imate mobitel za vrijeme kolokvija, kolokvij se poništava i pokreće se stegovni postupak protiv vas.

Zadatak 1 (13+2 bodova) Napišite funkciju imena `linearHanoi` koja za zadani broj diskova n, početni štap, konačni štap i pomoćni štap, ispisuje sve korake potrebne da diskove prebacimo s početnog na konačni štap, koristeći pomoćni štap i sljedeći standardna pravila problema Hanoi uz dodatak da diskove smijemo micati samo s jednog susjednog štapa na drugi. Dakle, za štapove $0, 1, 2$ numerirane s lijeva na desno, možemo raditi pomake $0 \to 1$, $1 \to 2$, $2 \to 1$ i $1 \to 0$ ukoliko navedeni pomaci ne krše druga pravila problema u danom trenutku. Funkcija treba spremiti broj pomaka u varijabilni argument `brojKoraka`. Napišite i glavni program u kojem pozivate funkciju za 3 diska, te ispisujete odgovarajući broj koraka.

Napomena: Nije dozvoljeno korištenje dodatnih polja niti biblioteka osim `stdio.h`.

```
void prebaci_jednog1(int o, int k){
          printf("Prebaci s %d na %d\n", o, k);
}

void linearHanoi1(int n, int o, int k, int p, int *brKoraka){

          if(n<=1){
              if(o<k){
                for(int i=o;i<k;i++) prebaci_jednog1(i,i+1);
                *brKoraka+=k-o;
              }
              else if(k<o){
                for(int i=o;i>k;i--) prebaci_jednog1(i,i-1);
                *brKoraka+=o-k;
              }
          }
          else{
                linearHanoi1(n-1,o,k,p,brKoraka);
                prebaci_jednog1(o,p);
                (*brKoraka)++;
                linearHanoi1(n-1,k,o,p,brKoraka);
                prebaci_jednog1(p,k);
                (*brKoraka)++;
                linearHanoi1(n-1,o,k,p, brKoraka);
          }
}

int main(void){
  int bb = 0;
  bb = linearHanoi1(3,0,2,1,&bb);
  printf("\nBroj koraka: %d\n",bb);

 return 0;

}
```

JMBAG

IME I PREZIME

Programiranje 2 – kolokvij, 2. 5. 2025.

Zadatak 2 (4+6 bodova) Neka je $X \in \mathbb{Z}^{n \times m}$ matrica cijelih brojeva. Za element `X[i][j]` (element matrice u retku i i stupcu j), kazemo da je **sedlasta točka** ako je maksimalan u svojem retku i minimalan u svojem stupcu. Npr. u matrici

$$X = \begin{bmatrix} 1 & 3 & -2 \\ -5 & -1 & -1 \end{bmatrix}$$

element `X[1][2]` je jedina sedlasta točka.

a) Napišite funkciju `jeSedlastaTocka` koja prima matricu $X \in \mathbb{Z}^{n \times m}$, njezine dimenzije n (broj redaka) i m (broj stupaca), te indekse retka i i stupca j. Funkcija treba vratiti 1 ako je element `X[i][j]` sedlasta točka. U suprotnom, funkcija treba vratiti 0.

b) Napišite funkciju `obradiNajvecuPodmatricuBezSedla` koja prima matricu cijelih brojeva `X` i njezine dimenzije n i m. Ova funkcija treba pronaći najveću kvadratnu podmatricu unutar `X` koja NE sadrži niti jednu točku sedla. "Najveća" podmatrica se definira prvo po dimenziji. Ako postoji više podmatrica iste najveće dimenzije koje ne sadrže sedlastu točku, uzima se ona čiji je gornji lijevi kut najviše (manji indeks retka), a zatim ona čiji je gornji lijevi kut najviše lijevo (manji indeks stupca). Ako je takva podmatrica pronađena, postavite sve njezine elemente na vrijednost 0. Funkcija treba vratiti dimenziju pronađene najveće kvadratne podmatrice bez sedlaste točke. Ako takva podmatrica ne postoji, funkcija vraća 0.

Napomena: Pretpostavite da su dimenzije matrice $n, m \leq 15$. Elementi matrice su cijeli brojevi. Matrica je 0-indeksirana. Nije potrebno provjeravati jesu li ulazne dimenzije unutar zadanih ograničenja. Jedina dozvoljena biblioteka u ovom zadatku je `stdio.h`.

```
#include <stdio.h>

int jeSedlastaTocka(int X[15][15], int n, int m, int i, int j) {
    int max_u_retku = 1;
    for (int c = 0; c < m; c++) {
        if (X[i][c] > X[i][j]) {
            max_u_retku = 0;
            break;
        }
    }

    int min_u_stupcu = 1;
    for (int r = 0; r < n; r++) {
        if (X[r][j] < X[i][j]) {
            min_u_stupcu = 0;
            break;
        }
    }

    return max_u_retku && min_u_stupcu;
}

// Pomocna funkcija provjerava je li d x d podmatrica s gornjim lijevim
// kantunom u X[gl_i][gl_j] sadrzi sedlastu tocku
int podmatImaSedlo(int X[15][15], int n, int m, int gl_i, int gl_j, int d) {
    for (int i = gl_i; i < gl_i + d; i++) {
        for (int j = gl_j; j < gl_j + d; j++) {
            if (jeSedlastaTocka(X, n, m, i, j)) {
                return 1;
            }
        }
    }
```

```
    return 0;
}

int obradiNajvecuPodmatricuBezSedla(int X[15][15], int n, int m) {
    // Ove var opisuju najvecu pronadenu podmatricu bez sedla
    int naj_d = 0;
    int naj_i = -1, naj_j = -1;

    // iteriraj kroz sve moguce gornje lijeve kutove
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            for (int d = 1; i + d <= n && j + d <= m; d++) {
                if (!podmatImaSedlo(X, n, m, i, j, d) && d > naj_d) {
                    // Redoslijed iteracije garantira da cemo ovako izabrati
                    // matricu s najvise gore pa najvise lijevim kutem.
                    naj_d = d;
                    naj_i = i;
                    naj_j = j;
                }
            }
        }
    }

    // Ako je najveca podmatrica bez sedla pronadena, postavi el u 0
    if (naj_d > 0) {
        for (int i = naj_i; i < naj_i + naj_d; i++) {
            for (int j = naj_j; j < naj_j + naj_d; j++) {
                X[i][j] = 0;
            }
        }
    }

    return naj_d;
}

int main() {
    int n, m;
    int X[15][15];

    // Ucitaj dimenzije matrice
    scanf("%d %d", & n, & m);

    // Ucitaj elemente matrice
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            scanf("%d", & X[i][j]);
        }
    }

    // Obradi matricu
    obradiNajvecuPodmatricuBezSedla(X, n, m);

    // Ispisi obradenu matricu
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            printf("%d ", X[i][j]);
        }
        printf("\n");
    }

    return 0;
}
```

JMBAG

IME I PREZIME

Programiranje 2 – kolokvij, 2. 5. 2025.

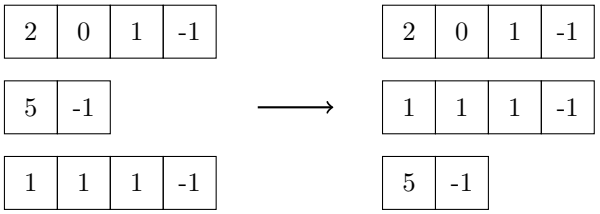
Zadatak 3 (5+ 8 (13) + 2 bodova) Svaki prirodni broj $n > 1$ se može rastaviti na proste faktore na sljedeći način:

$$n = p_1^{a_1} p_2^{a_2} \dots p_k^{a_k},$$

gdje su $p_1, p_2, \dots, p_k$ **prvih** k **prostih** brojeva, a eksponenti $a_1, a_2, \dots, a_{k-1}$ nenegativni (≥ 0) cijeli brojevi, pri čemu je $a_k \geq 1$ (npr. $15 = 2^0 3^1 5^1$, $49 = 2^0 3^0 5^0 7^2$).

a) Napišite funkciju `int nti_prosti(int n)` koja prima prirodan broj $n > 1$ te računa n-ti po redu prosti broj..

b) Napišite funkciju `void rastav_sort(int **brojevi, int m)` koja prima niz od m nizova nenegativnih cijelih brojeva koji predstavljaju eksponente u rastavu broja na gore opisani način. Predzadnji element je veći ili jednak 1, a zadnji element u svakom nizu je -1 te on označava kraj niza (nije dio rastava). Funkcija treba sortirati nizove (rastave) zamjenom pokazivača uzlazno po veličini broja kojeg pojedini niz predstavlja. Na primjer:



jer je $2^2 3^0 5^1 = 20$, $2^5 = 32$, $2^1 3^1 5^1 = 30$.

c) Napišite glavni program koji prvo učitava prirodan broj m, potom učitava m nizova prirodnih brojeva. Program treba sortirati nizove koristeći funkciju `rastav_sort`.

Napomene:

- U podzadatku (b) možete koristiti funkciju iz prethodnog podzadatka čak i ako ju niste napisali, također i u podzadatku (c).
- Zadatak nosi **15** bodova, međutim ukoliko se umjesto algoritma sortiranja složenosti $\mathcal{O}(n \log n)$ koristi algoritam složenosti $\mathcal{O}(n^2)$, zadatak nosi maksimalno **10** bodova. Korištenje quickSort algoritma iz biblioteke `stdlib.h` nosi dodatnih **5** bodova, odnosno tako riješen zadatak u sumi nosi **20** bodova.
- Svi nizovi korišteni u ovome zadatku moraju biti dinamički (re)alocirani (i na kraju dealocirani!), te moraju zauzimati točno onoliko memorije koliko je potrebno! Ne treba provjeravati uspješnost (re)alokacije. Uz standardnu biblioteku `stdio.h`, u ovome zadatku dozvoljeno je korištenje jedino biblioteke `stdlib.h`.

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

/* (a) podzadatak */
/*pomocna funkcija, ispituje je li prost*/
int prost(int i){
    for(int j = 2; j*j <= i; j++){
        if(i % j == 0) return 0;
    }
    return 1;
}

int nti_prosti(int n){
    int br = 1, p = 2;
    while(br < n){
        p++;
        if(prost(p)) br++;
    }
    return p;
```

```
}

/* (b) podzadatak */
/* funkcija za racunanje potencija */
int pwr(int p, int a){
    int pot = 1;
    for(int i = 1; i <= a; i++)
        pot *= p;
    return pot;
}

/* funkcija usporedbe za qsort */
int comp(const void *a, const void *b) {
    int *niz1 = *(int**)a;
    int *niz2 = *(int**)b;
    int n = 1, br1 = 1, br2 = 1;
    while(niz1[n-1] != -1){
        br1 *= pwr(nti_prosti(n), niz1[n-1]);
        n++;
    }
    n = 1;
    while(niz2[n-1] != -1){
        br2 *= pwr(nti_prosti(n), niz2[n-1]);
        n++;
    }
    return br1 - br2;
}

void rastav_sort(int **brojevi, int m){
    qsort(brojevi, m, sizeof(brojevi[0]), comp);
    return;
}

/* (c) podzadatak */
int main(void){
    int m, **brojevi, i;
    printf("Unesite m: "); scanf("%d", &m);
    brojevi = (int**)malloc(m * sizeof(int*));
    for(i = 0; i < m; i++){
        brojevi[i] = NULL;
        printf("Unesite %d. niz:\n", i+1);
        int d = 0;
        while(1){
            int x;
            scanf("%d", &x);
            brojevi[i] = realloc(brojevi[i], (++d)*sizeof(int));
            brojevi[i][d-1] = x;
            if(x == -1) break;
        }
    }
    rastav_sort(brojevi, m);
    for(i = 0; i < m; i++) free(brojevi[i]);
    free(brojevi);
}
```

JMBAG

IME I PREZIME

Programiranje 2 – kolokvij, 2. 5. 2025.

Zadatak 4 (5+5 bodova) U pripremi putovanja, putnička agencija treba organizirati podatke o putnicima. U nizu stringova `putnici` spremljeni su stringovi maksimalne duljine 40 u formatu `"Ime␣Prezime␣DD/MM/YYYY"`, gdje podstring oblika `"Ime␣Prezime"` označava ime i prezime putnika, a `"DD/MM/YYYY"` njegov datum rođenja.

Napišite funkciju s prototipom

`void izdvoji(char putnici[][41], int n, char izdvojeni[][41], char* datum)`,

gdje je `putnici` ranije opisan niz putnika, a `n` broj putnika u tom nizu. Funkcija treba u argument `izdvojeni` upisati stringove iz niza `putnici` koji odgovaraju putnicima koji su rođeni **prije** datuma upisanog u string `datum` također u formatu `"DD/MM/YYYY"`. Dodatno, funkcija provjerava jesu li u podstringu oblika `"Ime␣Prezime"` u stringovima niza `izdvojeni` upisani znakovi koji nisu slova. Svaki znak koji nije slovo, osim razmaka, funkcija mijenja s `'?'` te ako postoji barem jedan takav znak, sva slova treba promijeniti u velika. Potom funkcija treba redom ispisati stringove iz niza `izdvojeni`.

Primjer: Za funkciju `main`:

```
int main()
{
    char putnici[4][41], izdvojeni[4][41];

    strcpy(putnici[0], "Per-o Peric 09/10/2005");
    strcpy(putnici[1], "Ana Anic 22/12/2007");
    strcpy(putnici[2], "Vid Vi3d.ic 28/07/2005");
    strcpy(putnici[3], "Ma,ja Ma9jic 27/07/2005");

    izdvoji(putnici,4,izdvojeni,"30/07/2005");

    return 0;
}
```

ispis treba biti:

```
VID VI?D?IC 28/07/2005
MA?JA MA?JIC 27/07/2005
```

Napomena: Dozvoljeno je korištenje pomoćnih polja charova.

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

int stariji(char* putnik, char* datum) // ispitujemo je li putnik rodjen prije datuma
{
    char temp1[41], temp2[41];
    int duljina=strlen(putnik);

    // usporedba godina
    strcpy(temp1,putnik+duljina-4);
    strcpy(temp2,datum+6);

    if(strcmp(temp1,temp2)<0) return 1;

    if(strcmp(temp1,temp2)>0) return 0;

    // usporedba mjeseca, ako su iste godine
    strcpy(temp1,putnik+duljina-7);
    strcpy(temp2,datum+3);

    if(strcmp(temp1,temp2)<0) return 1;
```

```
    if(strcmp(temp1,temp2)>0) return 0;

    // usporedba dana, ako su isti mjeseci i godine
    strcpy(temp1,putnik+duljina-10);
    strcpy(temp2,datum);

    if(strcmp(temp1,temp2)<0) return 1;

    if(strcmp(temp1,temp2)>=0) return 0;
}

void izdvoji(char putnici[][41], int n, char izdvojeni[][41], char* datum)
{
    int i, j, flag, br=0, duljina;
    char c;

    for(i=0; i<n; i++)
        if(stariji(putnici[i],datum))
            strcpy(izdvojeni[br++],putnici[i]);

    for(i=0; i<br; i++)
    {
        flag=0;
        duljina=strlen(izdvojeni[i]);
        for(j=0; j<duljina-11; j++)
            if(!isalpha(izdvojeni[i][j]) && izdvojeni[i][j]!=' ')
            {
                flag=1;
                izdvojeni[i][j]='?';
            }
        if(flag)
            for(j=0; j<duljina-11; j++)
            {
                c=toupper(izdvojeni[i][j]);
                izdvojeni[i][j]=c;
            }
    }

    for(i=0; i<br; i++)
        printf("%s\n", izdvojeni[i]);
}

int main()
{
    char putnici[4][41], izdvojeni[4][41];

    strcpy(putnici[0], "Per-o Peric 09/10/2005");
    strcpy(putnici[1], "Ana Anic 22/12/2007");
    strcpy(putnici[2], "Vid Vi3d.ic 28/07/2005");
    strcpy(putnici[3], "Ma,ja Ma9jic 27/07/2005");

    izdvoji(putnici,4,izdvojeni,"30/07/2005");

    return 0;
}
```

0

JMBAG

IME I PREZIME

Programiranje 2 – kolokvij, 2. 5. 2025.

Napomene. Svako rješenje napišite isključivo na papir sa zadatkom jer jedino njega predajete. Pomoćne račune smijete raditi na drugim papirima koje će vam dati dežurni asistent. U svim zadacima **obavezno** pišite postupak!
Dozvoljeno je korištenje isključivo pribora za pisanje i brisanje, te službenog podsjetnika. Kalkulatori, te razne tablice, papiri i sl. nisu dozvoljeni! **Mobitele** isključite i pospremite daleko od sebe! Ako se ustanovi da **kod sebe** imate mobitel za vrijeme kolokvija, kolokvij se poništava i pokreće se stegovni postupak protiv vas.

Zadatak 1 (13+2 bodova) Napišite funkciju imena `linearHanoi` koja za zadani broj diskova n, početni štap, konačni štap i pomoćni štap, ispisuje sve korake potrebne da diskove prebacimo s početnog na konačni štap, koristeći pomoćni štap i sljedeći standardna pravila problema Hanoi uz dodatak da diskove smijemo micati samo s jednog susjednog štapa na drugi. Dakle, za štapove $0, 1, 2$ numerirane s lijeva na desno, možemo raditi pomake $0 \to 1$, $1 \to 2$, $2 \to 1$ i $1 \to 0$ ukoliko navedeni pomaci ne krše druga pravila problema u danom trenutku. Funkcija treba vratiti broj napravljenih pomaka kao povratnu vrijednost funkcije. Napišite i glavni program u kojem pozivate funkciju za 3 diska, te ispisujete odgovarajući broj koraka.

Napomena: Nije dozvoljeno korištenje dodatnih polja niti biblioteka osim `stdio.h`.

```
int prebaci_jednog2(int o, int k){
          printf("Prebaci s %d na %d\n", o, k);
          return 1;
}

int linearHanoi2(int n, int o, int k, int p, int brKoraka){

          if(n<=1){
              if(o<k){
                for(int i=o;i<k;i++) brKoraka+=prebaci_jednog2(i,i+1);
              }
              else if(k<o){
                for(int i=o;i>k;i--) brKoraka+=prebaci_jednog2(i,i-1);
              }
          }
          else{
                brKoraka = linearHanoi2(n-1,o,k,p,brKoraka);
                brKoraka+=prebaci_jednog2(o,p);
                brKoraka = linearHanoi2(n-1,k,o,p,brKoraka);
                brKoraka+=prebaci_jednog2(p,k);
                brKoraka = linearHanoi2(n-1,o,k,p, brKoraka);
          }
          return brKoraka;
}

int main(void){
  bb = 0;
  printf("\nBroj koraka: %d",linearHanoi2(5,0,2,1,bb));

 return 0;

}
```

JMBAG

IME I PREZIME

Programiranje 2 – kolokvij, 2. 5. 2025.

Zadatak 2 (4+6 bodova) Neka je $X \in \mathbb{Z}^{n \times m}$ matrica cijelih brojeva. Za element `X[i][j]` (element matrice u retku i i stupcu j), kazemo da je **sedlasta točka** ako je minimalan u svojem retku i maksimalan u svojem stupcu. Npr. u matrici

$$X = \begin{bmatrix} 2 & 1 & -3 \\ 3 & 2 & 6 \end{bmatrix}$$

element `X[1][2]` je jedina sedlasta točka.

a) Napišite funkciju `jeSedlastaTocka` koja prima matricu $X \in \mathbb{Z}^{n \times m}$, njezine dimenzije n (broj redaka) i m (broj stupaca), te indekse retka i i stupca j. Funkcija treba vratiti 1 ako je element `X[i][j]` sedlasta točka. U suprotnom, funkcija treba vratiti 0.

b) Napišite funkciju `obradiNajvecuPodmatricuBezSedla` koja prima matricu cijelih brojeva `X` i njezine dimenzije n i m. Ova funkcija treba pronaći najveću kvadratnu podmatricu unutar `X` koja NE sadrži niti jednu točku sedla. "Najveća" podmatrica se definira prvo po dimenziji. Ako postoji više podmatrica iste najveće dimenzije koje ne sadrže sedlastu točku, uzima se ona čiji je gornji lijevi kut najniže (veći indeks retka), a zatim ona čiji je gornji lijevi kut najviše desno (veći indeks stupca). Ako je takva podmatrica pronađena, postavite sve njezine elemente na vrijednost -1. Funkcija treba vratiti dimenziju pronađene najveće kvadratne podmatrice bez sedlaste točke. Ako takva podmatrica ne postoji, funkcija vraća 0.

Napomena: Pretpostavite da su dimenzije matrice $n, m \leq 15$. Elementi matrice su cijeli brojevi. Matrica je 0-indeksirana. Nije potrebno provjeravati jesu li ulazne dimenzije unutar zadanih ograničenja. Jedina dozvoljena biblioteka u ovom zadatku je `stdio.h`.

```
#include <stdio.h>

int jeSedlastaTocka(int X[15][15], int n, int m, int i, int j) {
    int min_u_retku = 1;
    for (int c = 0; c < m; c++) {
        if (X[i][c] < X[i][j]) {
            min_u_retku = 0;
            break;
        }
    }

    int max_u_stupcu = 1;
    for (int r = 0; r < n; r++) {
        if (X[r][j] > X[i][j]) {
            max_u_stupcu = 0;
            break;
        }
    }

    return min_u_retku && max_u_stupcu;
}

// Pomocna funkcija provjerava je li d x d podmatrica s gornjim lijevim
// kantunom u X[gl_i][gl_j] sadrzi sedlastu tocku
int podmatImaSedlo(int X[15][15], int n, int m, int gl_i, int gl_j, int d) {
    for (int i = gl_i; i < gl_i + d; i++) {
        for (int j = gl_j; j < gl_j + d; j++) {
            if (jeSedlastaTocka(X, n, m, i, j)) {
                return 1;
            }
        }
    }
```

```
    return 0;
}

int obradiNajvecuPodmatricuBezSedla(int X[15][15], int n, int m) {
    // Ove var opisuju najvecu pronadenu podmatricu bez sedla
    int naj_d = 0;
    int naj_i = -1, naj_j = -1;

    // iteriraj kroz sve moguce gornje lijeve kutove
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            for (int d = 1; i + d <= n && j + d <= m; d++) {
                if (!podmatImaSedlo(X, n, m, i, j, d) && d >= naj_d) {
                    // Redoslijed iteracije garantira da cemo ovako izabrati
                    // matricu s najvise dolje pa najvise desnim kutem.
                    naj_d = d;
                    naj_i = i;
                    naj_j = j;
                }
            }
        }
    }

    // Ako je najveca podmatrica bez sedla pronadena, postavi el u -1
    printf("Naj: %d %d %d\n", naj_i ,naj_j ,naj_d);

    if (naj_d > 0) {
        for (int i = naj_i; i < naj_i + naj_d; i++) {
            for (int j = naj_j; j < naj_j + naj_d; j++) {
                X[i][j] = -1;
            }
        }
    }

    return naj_d;
}

int main() {
    int n, m;
    int X[15][15];

    // Ucitaj dimenzije matrice
    scanf("%d %d", & n, & m);

    // Ucitaj elemente matrice
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            scanf("%d", & X[i][j]);
        }
    }

    // Obradi matricu
    obradiNajvecuPodmatricuBezSedla(X, n, m);

    // Ispisi obradenu matricu
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            printf("%d ", X[i][j]);
        }
        printf("\n");
    }
```

```
    return 0;
}
```

JMBAG

IME I PREZIME

Programiranje 2 – kolokvij, 2. 5. 2025.

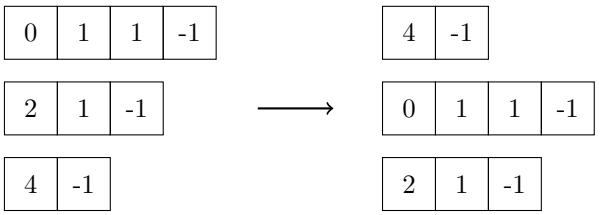
Zadatak 3 (5 + 8 (13) + 2 bodova) Svaki prirodni broj $m > 1$ se može rastaviti na proste faktore na sljedeći način:

$$m = q_1^{\beta_1} q_2^{\beta_2} \dots q_k^{\beta_k},$$

gdje su $q_1, q_2, \dots, q_k$ **prvih** k **prostih** brojeva, a eksponenti $\beta_1, \beta_2, \dots, \beta_{k-1}$ nenegativni (≥ 0) cijeli brojevi, pri čemu je $\beta_k \geq 1$ (npr. $14 = 2^1 3^0 5^0 7^1$, $25 = 2^0 3^0 5^2$).

a) Napišite funkciju `int iti_prosti(int i)` koja prima prirodan broj $i \geq 1$ te računa i-ti po redu prosti broj.

b) Napišite funkciju `void OsnTeoremAritm_sort(int **rastavi, int n)` koja prima niz od n nizova nenegativnih cijelih brojeva koji predstavljaju eksponente u rastavu broja na gore opisani način. Predzadnji element je veći ili jednak 1, a zadnji element u svakom nizu je -1 te on označava kraj niza (nije dio rastava). Funkcija treba sortirati nizove (rastave) zamjenom pokazivača silazno po veličini broja kojeg pojedini niz predstavlja. Na primjer:



jer je $2^0 3^1 5^1 = 15$, $2^4 = 16$, $2^2 3^1 = 12$.

c) Napišite glavni program koji prvo učitava prirodan broj n, potom učitava n nizova prirodnih brojeva. Program treba sortirati nizove koristeći funkciju `OsnTeoremAritm_sort`.

Napomene:

- U podzadatku (b) možete koristiti funkciju iz prethodnog podzadatka čak i ako ju niste napisali, također i u podzadatku (c).
- Zadatak nosi **15** bodova, međutim ukoliko se umjesto algoritma sortiranja složenosti $\mathcal{O}(n \log n)$ koristi algoritam složenosti $\mathcal{O}(n^2)$, zadatak nosi maksimalno **10** bodova. Korištenje quickSort algoritma iz biblioteke `stdlib.h` nosi dodatnih **5** bodova, odnosno tako riješen zadatak u sumi nosi **20** bodova.
- Svi nizovi korišteni u ovome zadatku moraju biti dinamički (re)alocirani (i na kraju dealocirani!), te moraju zauzimati točno onoliko memorije koliko je potrebno! Ne treba provjeravati uspješnost (re)alokacije. Uz standardnu biblioteku `stdio.h`, u ovome zadatku dozvoljeno je korištenje jedino biblioteke `stdlib.h`.

JMBAG

IME I PREZIME

Programiranje 2 – kolokvij, 2. 5. 2025.

Zadatak 4 (5+5 bodova) U pripremi školskog izleta, nastavnici trebaju organizirati podatke o učenicima. U nizu stringova `ucenici` spremljeni su stringovi maksimalne duljine 30 u formatu `"Ime_Prezime_DD/MM/YYYY"`, gdje podstring oblika `"Ime_Prezime"` označava ime i prezime učenika, a `"DD/MM/YYYY"` njegov datum rođenja.

Napišite funkciju s prototipom

`void izdvoji(char ucenici[][31], int n, char izdvojeni[][31], char* datum)`,

gdje je `ucenici` ranije opisan niz učenika, a `n` broj učenika u tom nizu. Funkcija treba u argument `izdvojeni` upisati stringove iz niza `ucenici` koji odgovaraju učenicima koji su rođeni **prije** datuma upisanog u string `datum` također u formatu `"DD/MM/YYYY"`. Dodatno, funkcija provjerava jesu li u podstringu oblika `"Ime_Prezime"` u stringovima niza `izdvojeni` upisani znakovi koji nisu slova. Svaki znak koji nije slovo, osim razmaka, funkcija mijenja s `'*'` te ako postoji barem jedan takav znak, sva slova treba promijeniti u mala. Potom funkcija treba redom ispisati stringove iz niza `izdvojeni`.

Primjer: Za funkciju `main`:

```
int main()
{
    char ucenici[4][31], izdvojeni[4][31];

    strcpy(ucenici[0], "L2uka Luk=ic 10/12/2010");
    strcpy(ucenici[1], "1Iva Iv-ic 05/05/2006");
    strcpy(ucenici[2], "Mar)ko Markic 12/11/2012");
    strcpy(ucenici[3], "Tena, Ten8&ic 14/12/2010");

    izdvoji(ucenici,4,izdvojeni,"15/12/2010");

    return 0;
}
```

ispis treba biti:

```
l*uka luk*ic 10/12/2010
*iva iv*ic 05/05/2006
tena* ten**ic 14/12/2010
```

Napomena: Dozvoljeno je korištenje pomoćnih polja charova.

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

int stariji(char* ucenik, char* datum) // ispitujemo je li putnik rodjen prije datuma
{
    char temp1[31], temp2[31];
    int duljina=strlen(ucenik);

    // usporedba godina
    strcpy(temp1,ucenik+duljina-4);
    strcpy(temp2,datum+6);

    if(strcmp(temp1,temp2)<0) return 1;

    if(strcmp(temp1,temp2)>0) return 0;

    // usporedba mjeseca, ako su iste godine
    strcpy(temp1,ucenik+duljina-7);
    strcpy(temp2,datum+3);

    if(strcmp(temp1,temp2)<0) return 1;
```

```
    if(strcmp(temp1,temp2)>0) return 0;

    // usporedba dana, ako su isti mjeseci i godine
    strcpy(temp1,ucenik+duljina-10);
    strcpy(temp2,datum);

    if(strcmp(temp1,temp2)<0) return 1;

    if(strcmp(temp1,temp2)>=0) return 0;
}

void izdvoji(char ucenici[][31], int n, char izdvojeni[][31], char* datum)
{
    int i, j, flag, br=0, duljina;
    char c;

    for(i=0; i<n; i++)
        if(stariji(ucenici[i],datum))
            strcpy(izdvojeni[br++],ucenici[i]);

    for(i=0; i<br; i++)
    {
        flag=0;
        duljina=strlen(izdvojeni[i]);
        for(j=0; j<duljina-11; j++)
            if(!isalpha(izdvojeni[i][j]) && izdvojeni[i][j]!=' ')
            {
                flag=1;
                izdvojeni[i][j]='*';
            }
        if(flag)
            for(j=0; j<duljina-11; j++)
            {
                c=tolower(izdvojeni[i][j]);
                izdvojeni[i][j]=c;
            }
    }

    for(i=0; i<br; i++)
        printf("%s\n", izdvojeni[i]);
}

int main()
{
    char ucenici[4][31], izdvojeni[4][31];

    strcpy(ucenici[0], "L2uka Luk=ic 10/12/2010");
    strcpy(ucenici[1], "1Iva Iv-ic 05/05/2006");
    strcpy(ucenici[2], "Mar)ko Markic 12/11/2012");
    strcpy(ucenici[3], "Tena, Ten8&ic 14/12/2010");

    izdvoji(ucenici,4,izdvojeni,"15/12/2010");

    return 0;
}
```