Are you sure you want to submit the exam?

No, continue writing Yes, submit my exam!

My answers:

1

S tipkovnice učitati cijele brojeve `N`, `M`, `idxi` i `idxj`. Nije potrebno provjeravati njihovu ispravnost.
Zatim inicijalizirati dvodimenzijsko polje `polje` veličine `N(redaka) x M(stupaca)` tako da nakon inicijalizacije svaki član polja ima vrijednost `0`.

Potom, počevši od člana `polje[idxi][idxj]` popuniti matricu `polje` prema sljedećim pravilima:

- Ako je vrijednost trenutnog člana `polje[idxi][idxj]` jednaka `1`, popunjavanje matrice je gotovo
- Ako je vrijednost trenutnog člana `polje[idxi][idxj]` jednaka `0`, postavite mu vrijednost na `1`,
  te se pomaknite po retcima za `idxi = idxi + i_smjer` i po stupcima za `idxj = idxj + j_smjer`
- Ako bi kretanje rezultiralo izlaskom iz matrice tj. ako bi `idxi = idxi + i_smjer` ili `idxj = idxj + j_smjer` rezultirali izlaskom iz matrice, smjer kretanja se onda mijenja na sljedeći način:
- Kada stignemo do **donjeg** ili **gornjeg** ruba matrice, počinjemo se kretati po retcima u suprotnom smjeru (`-i_smjer`)
- Kada stignemo do **desnog** ili **lijevog** ruba matrice, počinjemo se kretati po stupcima u suprotnom smjeru (`-j_smjer`)

My answer:

```
#include <stdio.h>

int main() {
    int N, M, idxi, idxj, i_smjer = -1, j_smjer = 1, duljina = 0;
    printf("Upisite brojeve N, M, idxi i idxj > ");
    scanf("%d %d %d %d", &N, &M, &idxi, &idxj);
    int polje[N][M];

    for(int i = 0; i < N; i++)
        for(int j = 0; j < M; j++)
            polje[i][j] = 0;

    while(polje[idxi][idxj] == 0) {
        if(i_smjer == -1 && idxi == 0)
            i_smjer *= -1;
        else if(i_smjer == 1 && idxi == N - 1)
            i_smjer *= -1;
        if(j_smjer == 1 && idxj == M - 1)
            j_smjer *= -1;
        else if(j_smjer == -1 && idxj == 0)
            j_smjer *= -1;
        polje[idxi][idxj] = 1;
        idxi += i_smjer;
        idxj += j_smjer;
        duljina++;
    }

    for(int i = 0; i < N; i++) {
        for(int j = 0; j < M; j++)
            if(j == 0)
                printf("%d", polje[i][j]);
            else
                printf("%2d", polje[i][j]);
        printf("\n");
    }
    printf("Duljina: %d\n", duljina);

    return 0;
}
```

2

Napišite program koji će učitati prirodni broj `n` iz intervala `[1, 10]`. Broj `n` predstavlja dimenzije kvadratne matricu (`n x n`). Nadalje, program treba učitati vrijednosti matrice po retcima.

Za svaki stupac, potrebno je ispisati razliku suma trenutnog stupca i sljedećeg stupca. Od sume zadnjeg stupca potrebno je oduzeti sumu prvog stupca.

Nije potrebno provjeravati ispravnost unesenih vrijednosti.

Napomena: dvodimenzijsko polje definirati tako da se za njegovu pohranu troši minimalni potreban prostor.

Primjer izvođenja programa:

```
Unesite·n·>·3↵
```

My answer:

```
#include <stdio.h>

int main() {
    int n;
    printf("Unesite n > ");
    scanf("%d", &n);
    int polje[n][n];
    int suma[n];
    for(int i = 0; i < n; i++)
        suma[i] = 0;

    for(int i = 0; i < n; i++)
        for(int j = 0; j < n; j++)
            scanf("%d", &polje[i][j]);

    for(int j = 0; j < n; j++)
        for(int i = 0; i < n; i++)
            suma[j] += polje[i][j] - polje[i][(j+1)%(n)];

    printf("Rezultat > ");
    for(int i = 0; i < n; i++)
        if(i == n-1)
            printf("%d\n", suma[i]);
        else
            printf("%d ", suma[i]);
}
```

3

S tipkovnice učitati niz znakova (string) koji sigurno neće biti dulji od 200 znakova.
Niz može sadržavati bilo koji znak ASCII tablice koji se može tiskati (pozicije 32-126).
Odredite i na zaslon ispišite obrnuti niz i pri tome sva mala slova pretvorite u velika.

Primjeri izvršavanja programa:

```
Upisite·znakovni·niz·>·PrImJer↵
Obrnuti·niz:·REJMIRP↵

Upisite·znakovni·niz·>·Ovo·je·Testni·primjer99↵
Obrnuti·niz:·99RJEMIRP·INTSET·EJ·OVO↵
```

My answer:

```
#include <stdio.h>
#define MAX 200

int main() {
    char polje[MAX], duljina = 0;
    printf("Upisite znakovni niz > ");
    fgets(polje, 200, stdin);

    for(int i = 0; polje[i] != '\0'; i++) {
        duljina++;
        if(polje[i] >= 'a' && polje[i] <= 'z')
            polje[i] += ('A' - 'a');
    }

    printf("Obrnuti niz: ");
    for(int i = duljina - 1; i >= 0; i--)
        printf("%c", polje[i]);
    printf("\n");

    return 0;
}
```

4

Napišite program koji će za dvodimenzijsko polje cijelih brojeva s jednakim brojem redaka i stupaca N (kvadratnu matricu) izračunati sumu svih brojeva koji se nalaze u zadanom retku i stupcu. Element matrice u kojem se zadani redak i stupac "sijeku" zbraja se samo jedanput.

S tipkovnice program treba učitati prirodni broj N. Zatim je potrebno s tipkovnice učitati po retcima članove kvadratne matrice. Na kraju je potrebno učitati redni broj retka i stupca čije elemente zbrajamo (0 je redni broj prvog retka/stupca). Nije potrebno provjeravati ispravnost unesenih vrijednosti.

Napomena: dvodimenzijsko polje definirati tako da se za njegovu pohranu troši minimalni potreban prostor.

Primjeri izvršavanja programa:

```
Upisite·broj·redaka/stupaca·matrice·>·3↵
```

My answer:

```
#include <stdio.h>

int main() {
    int N, redak, stupac, suma = 0;
    printf("Upisite broj redaka/stupaca matrice > ");
    scanf("%d", &N);
    int polje[N][N];
    printf("Upisite %dx%d clanova > \n", N, N);
    for(int i = 0; i < N; i++)
        for(int j = 0; j < N; j++)
            scanf("%d", &polje[i][j]);
    printf("Upisite redni broj retka > ");
    scanf("%d", &redak);
    printf("Upisite redni broj stupca > ");
    scanf("%d", &stupac);

    for(int j = 0; j < N; j++)
        suma += polje[redak][j];
    for(int i = 0; i < N; i++)
        if(i != redak)
            suma += polje[i][stupac];
    printf("Suma brojeva koji se nalaze u retku %d i stupcu %d je %d.", redak, stupac, suma);
    return 0;
}
```

5

S tipkovnice učitati niz znakova (string) koji sigurno neće biti dulji od 150 znakova.
Niz može sadržavati bilo koji znak ASCII tablice koji se može tiskati (pozicije 32-126).
Treba napisati program koji će ukloniti sve znakove koji su uneseni, a koji nisu dio engleske abecede. Ukoliko ulazni string ne sadrži niti jedan znak engleskog alfabeta treba ispisati tekst `prazan string`.

Primjeri izvršavanja programa:

```
Upisite·znakovni·niz·>·p2'r-o@gram84iz./↵
Izlazni·znakovni·niz·je·:·programiz↵

Upisite·znakovni·niz·>·12321[]↵
Izlazni·znakovni·niz·je·:·prazan·string↵
```

My answer:

```
#include <stdio.h>
#define MAX 150

int main() {
    char polje[MAX];
    _Bool prazan = 1;
    printf("Upisite znakovni niz > ");
    fgets(polje, MAX, stdin);

    printf("Izlazni znakovni niz je : ");
    for(int i = 0; polje[i] != '\0'; i++)
        if((polje[i] >= 'a' && polje[i] <= 'z') || (polje[i] >= 'A' && polje[i] <= 'Z')) {
            prazan = 0;
            printf("%c", polje[i]);
        }
    if(prazan) {
        printf("prazan string\n");
        return 0;
    }

    printf("\n");
    return 0;
}
```

6

S tipkovnice je potrebno učitati niz znakova (`string`) `niz`. Niz neće biti dulji od `50` znakova te ga ne treba provjeravati.

Potrebno je ispisati koji se znak najčešće pojavljuje u nizu `niz` te koliko se puta ponovio taj znak.

```
Znak·"A"·se·ponavlja·5·puta.
```

Ukoliko postoji više znakova koji se ponavljaju jednako puta, tada je potrebno ispisati sve znakove, te njihov broj ponavljanja. Ispis znakova treba biti takav da se prvo ispisuje onaj znak koji ima **najmanju vrijednost u ASCII tablici** pa nakon njega znak sa sljedećom najmanjom vrijednosti i tako dalje.

```
Znakovi·"A",·"C",·"a"·se·ponavljaju·3·puta.
```

My answer:

```
#include <stdio.h>
#define MAX 50
#define ASCII_MAX 256

int main() {
    char polje[MAX], znakovi[ASCII_MAX] = { 0 };
    int najveci = 0, brNajvecih = 0;
    _Bool prvi = 0;

    printf("Unesite niz > ");
    fgets(polje, MAX, stdin);

    for(int i = 0; polje[i] != '\0'; i++)
        znakovi[(int)polje[i]]++;

    for(int i = 0; i < ASCII_MAX; i++) {
        if(znakovi[i] == 0)
            continue;
        if(znakovi[i] > znakovi[najveci])
            najveci = i;
        else if(znakovi[i] == znakovi[najveci])
            brNajvecih++;
    }

    if(brNajvecih >= 1)
        printf("Znakovi");
    else
        printf("Znak");

    for(int i = 0; i < ASCII_MAX; i++) {
        if(znakovi[i] == 0 || znakovi[i] < znakovi[najveci])
            continue;

        if(brNajvecih >= 1) {
            if(!prvi) {
                printf(" \"%c\"", (char)i);
                prvi = 1;
                continue;
            }
            printf(", \"%c\"", (char)i);
            continue;
        }
        printf(" \"%c\"", (char)i);

    }
    if(brNajvecih >= 1) {
        printf(" se ponavljaju %d puta.\n", znakovi[najveci]);
        return 0;
    }

    printf(" se ponavlja %d puta.\n", znakovi[najveci]);

    return 0;
}
```

7

S tipkovnice učitati cijele brojeve `N`, `M`, `idxi` i `idxj`. Nije potrebno provjeravati njihovu ispravnost.
Zatim inicijalizirati dvodimenzijsko polje `polje` veličine `N(redaka) x M(stupaca)` tako da nakon inicijalizacije svaki član polja ima vrijednost `0`.

Potom, počevši od člana `polje[idxi][idxj]` popuniti matricu `polje` prema sljedećim pravilima:

- Ako je vrijednost trenutnog člana `polje[idxi][idxj]` jednaka `1`, popunjavanje matrice je gotovo
- Ako je vrijednost trenutnog člana `polje[idxi][idxj]` jednaka `0`, postavite mu vrijednost na `1`,
  te se pomaknite po retcima za `idxi = idxi + i_smjer` i po stupcima za `idxj = idxj + j_smjer`
- Ako bi kretanje rezultiralo izlaskom iz matrice tj. ako bi `idxi = idxi + i_smjer` ili `idxj = idxj + j_smjer` rezultirali izlaskom iz matrice, smjer kretanja se mijenja na sljedeći način:
- Kada stignemo do **donjeg** ili **gornjeg** ruba matrice, počinjemo se kretati po retcima u suprotnom smjeru (`-i_smjer`)
- Kada stignemo do **desnog** ili **lijevog** ruba matrice, počinjemo se kretati po stupcima u suprotnom smjeru (`-j_smjer`)

My answer:

```
#include <stdio.h>

int main() {
    int N, M, idxi, idxj, i_smjer = 1, j_smjer = 1, duljina = 0;
    printf("Upisite brojeve N, M, idxi i idxj > ");
    scanf("%d %d %d %d", &N, &M, &idxi, &idxj);
    int polje[N][M];

    for(int i = 0; i < N; i++)
        for(int j = 0; j < M; j++)
            polje[i][j] = 0;

    while(polje[idxi][idxj] == 0) {
        if(i_smjer == -1 && idxi == 0)
            i_smjer *= -1;
        else if(i_smjer == 1 && idxi == N - 1)
            i_smjer *= -1;
        if(j_smjer == 1 && idxj == M - 1)
            j_smjer *= -1;
        else if(j_smjer == -1 && idxj == 0)
            j_smjer *= -1;
        polje[idxi][idxj] = 1;
        idxi += i_smjer;
        idxj += j_smjer;
        duljina++;
    }

    for(int i = 0; i < N; i++) {
        for(int j = 0; j < M; j++)
            if(j == 0)
                printf("%d", polje[i][j]);
            else
                printf("%2d", polje[i][j]);
        printf("\n");
    }
    printf("Duljina: %d\n", duljina);

    return 0;
}
```

8

S tipkovnice učitati cijele brojeve `N`, `M`, `idxi` i `idxj`. Nije potrebno provjeravati njihovu ispravnost.
Zatim inicijalizirati dvodimenzijsko polje `polje` veličine `N(redaka) x M(stupaca)` tako da nakon inicijalizacije svaki član polja ima vrijednost `0`.

Potom, počevši od člana `polje[idxi][idxj]` popuniti matricu `polje` prema sljedećim pravilima:

- Ako je vrijednost trenutnog člana `polje[idxi][idxj]` jednaka `1`, popunjavanje matrice je gotovo
- Ako je vrijednost trenutnog člana `polje[idxi][idxj]` jednaka `0`, postavite mu vrijednost na `1`,
  te se pomaknite po retcima za `idxi = idxi + i_smjer` i po stupcima za `idxj = idxj + j_smjer`
- Ako bi kretanje rezultiralo izlaskom iz matrice tj. ako bi `idxi = idxi + i_smjer` ili `idxj = idxj + j_smjer` rezultirali izlaskom iz matrice, smjer kretanja se mijenja na sljedeći način:
- Kada stignemo do **donjeg** ili **gornjeg** ruba matrice, počinjemo se kretati po retcima u suprotnom smjeru (`-i_smjer`)
- Kada stignemo do **desnog** ili **lijevog** ruba matrice, počinjemo se kretati po stupcima u suprotnom smjeru (`-j_smjer`)

My answer:

```
#include <stdio.h>

int main() {
    int N, M, idxi, idxj, i_smjer = 1, j_smjer = -1, duljina = 0;
    printf("Upisite brojeve N, M, idxi i idxj > ");
    scanf("%d %d %d %d", &N, &M, &idxi, &idxj);
    int polje[N][M];

    for(int i = 0; i < N; i++)
        for(int j = 0; j < M; j++)
            polje[i][j] = 0;

    while(polje[idxi][idxj] == 0) {
        if(i_smjer == -1 && idxi == 0)
            i_smjer *= -1;
        else if(i_smjer == 1 && idxi == N - 1)
            i_smjer *= -1;
        if(j_smjer == 1 && idxj == M - 1)
            j_smjer *= -1;
        else if(j_smjer == -1 && idxj == 0)
            j_smjer *= -1;
        polje[idxi][idxj] = 1;
        idxi += i_smjer;
        idxj += j_smjer;
        duljina++;
    }

    for(int i = 0; i < N; i++) {
        for(int j = 0; j < M; j++)
            if(j == 0)
                printf("%d", polje[i][j]);
            else
                printf("%2d", polje[i][j]);
        printf("\n");
    }
    printf("Duljina: %d\n", duljina);

    return 0;
}
```

9

Napisati program koji pronalazi najveću sumu kontinuiranog pod-polja unutar zadanog polja cijelih brojeva te ispisuje sumu te sadržaj pronađenog pod-polja.

S tipkovnice treba učitati vrijednosti za veličinu jednodimenzionalnog polja `m`. Ne treba provjeravati jesu li upisane ispravne vrijednosti. Potrebno je učitati vrijednosti elemanata polja, te ispisati sadržaj pronađenog pod-polja kao i iznos najveće sume pod-polja. Pogledati primjere niže za pojašnjenja (nisu primjeri izvršavanja programa):

```
Polje od 9 članova {8 3 8 -5 4 3 -4 3 5}
ima najveću sumu kada se zbroji svih 9 članova: 25

Polje od 9 članova {8 3 8 -5 4 3 -4 3 -2}
ima najveću sumu kada se zbroji prvih 6 članova: 21

Polje od 12 članova {38 3 -50 100 200 300 400 500 600 700 800 900}
```

My answer:

```
#include <stdio.h>

enum tipovi {
    SVI = 0,
    PRVIH = 1,
    OD_DO = 2
};

struct tip{
    int id_pocetak;
    int id_kraj;
    int suma;
};

int main() {
    int m, najveca_suma = 0, najveci_tip = 0;
    struct tip tipovi[3];
    printf("Upisite velicinu polja m > ");
    scanf("%d", &m);
    int polje[m];

    printf("Upisite %d clanova polja m > ", m);
    for(int i = 0; i < m; i++)
        scanf("%d", &polje[i]);

    for(int i = 0; i < m; i++)
        najveca_suma += polje[i];

    tipovi[SVI].suma = najveca_suma;
    najveca_suma = 0;
    for(int i = 0; i < m; i++) {
        int suma = 0, j;
        for(j = 0; j < m - i; j++)
            suma += polje[j];
        if(suma > najveca_suma) {
            najveca_suma = suma;
            tipovi[PRVIH].id_kraj = j;
        }
    }
    tipovi[PRVIH].suma = najveca_suma;

    najveca_suma = 0;
    for(int i = 0; i < m; i++) {
        int suma = 0, j, k;
        for(j = m; j >= i; j--) {
            suma = 0;
            for(k = i; k < j; k++)
                suma += polje[k];
            if(suma > najveca_suma) {
                najveca_suma = suma;
                tipovi[OD_DO].id_pocetak = i;
                tipovi[OD_DO].id_kraj = j;
            }
        }

    }
    tipovi[OD_DO].suma  = najveca_suma;

    for(int i = 0; i < 3; i++)
        if(tipovi[i].suma > tipovi[najveci_tip].suma)
            najveci_tip = i;
    printf("Pronadjeno pod-polje:");
    switch(najveci_tip) {
        case SVI: {
            for(int i = 0; i < m; i++)
                printf(" %d", polje[i]);
            printf("\nNajveca suma kontinuiranog pod-polja je: %d", tipovi[SVI].suma);
            break;
        }
        case PRVIH: {
            for(int i = 0; i < tipovi[PRVIH].id_kraj; i++)
                printf(" %d", polje[i]);
            printf("\nNajveca suma kontinuiranog pod-polja je: %d", tipovi[PRVIH].suma);
            break;
        }
        case OD_DO: {
            for(int i = tipovi[OD_DO]. id_pocetak; i < tipovi[OD_DO].id_kraj; i++)
                printf(" %d", polje[i]);
            printf("\nNajveca suma kontinuiranog pod-polja je: %d", tipovi[OD_DO].suma);
            break;
        }
    }
    return 0;
}
```