# s

კურსის სახელწოდება: სტრუქტურული დაპროგრამება

ლექტორი: ნონა ოთხოზორია

ამოცანა 1

1. დაწერეთ პროგრამა, რომელიც უზრუნველყოფს კლავიატურიდან რამდენიმე რიცხვის შეტანას. შეტანილ რიცხვებს შორის, ის რიცხვები, რომლებიც 7-ზე გაყოფისას ნაშთში გვაძლევს 1, 2 ან 5-ს გადაწერეთ მასივში. მონაცემების შეტანა დასრულდეს, თუ შეტანილი მნიშვნელობა იქნება 0.

შეფასების რუბრიკა

 მონაცემების შეტანის ციკლური ორგანიზება - 2ქულა

 მასივის შექმნა - 3 ქულა

 ციკლიდან გამოსვლის ორგანიზება და შედეგის მიღება , გაუშვით პროგრამა შესრულებაზე და პროგრამის კოდთან ერთად გამოაგზავნეთ პროგრამის შესრულების შედეგი - სურათის სახით. - 2 ქულა.

```

#include <stdio.h>

int main() {

int array[100];

int index;

int input;

do {

printf("Enter a number: ");

scanf("%d", &input);

if(input % 7 == 1 || input % 7 == 2 || input % 7 == 5){

array[index] = input;

index ++;

}

} while (input != 0);

if (index > 0) {

printf("Array items: ");

for (int i = 0; i < index; i++) {

printf("%d ", array[i]);

}

printf("\n");

} else {

printf("No itmes in array.\n");

}

return 0;

}

```

ამოცანა 2

2. მოცემულია 10 ელემენტიანი მასივი

A[20]={3,4,6,7,12,15,17,18,11,24,15,-9,-20,-8,-10,-16]

დათვალეთ მოცემულ მასივში მარტივი რიცხვების რაოდენობა და გადაწერეთ ეს რიცხვები ახალ b\_ar მასივში.

A მასივში მარტივი რიცხვები ჩაანაცვლეთ 0-ებით.

შეფასების რუბრიკა

 მარტივი რიცხვების რაოდენობის დადგენა - 2 ქულა

 ახალი მასივის შექმნა - 3 ქულა

 „მარტივი“ რიცხვების ჩანაცვლება 0-ით, , გაუშვით პროგრამა შესრულებაზე და პროგრამის კოდთან ერთად გამოაგზავნეთ პროგრამის შესრულების შედეგი - 2 ქულა.

```

#include <stdio.h>

#include <stdbool.h>

bool isPrime(int num) {

if (num <= 1) {

return false;

}

for (int i = 2; i \* i <= num; i++) {

if (num % i == 0) {

return false;

}

}

return true;

}

int main() {

int A[20]={3,4,6,7,12,15,17,18,11,24,15,-9,-20,-8,-10,-16, 5,3,8,5};

int b\_ar[20];

int primeCount = 0;

for (int i = 0; i < 20; i++) {

if (isPrime(A[i])) {

b\_ar[primeCount] = A[i];

primeCount++;

A[i] = 0;

}

}

printf("Prime numbers in array A: ");

for (int i = 0; i < primeCount; i++) {

printf("%d ", b\_ar[i]);

}

printf("\n");

printf("Array A after replacing prime numbers with 0s: \n");

for (int i = 0; i < 20; i++) {

printf("%d ", A[i]);

}

return 0;

}

```

ამოცანა 3

შექმენით ორგანზომილებიანი მასივი A(4X4), მასივის პირველი სტრიქონი შეავსეთ 1-დან 10-მდე მარტივი რიცხვებით, მეორე სტრიქონი - 1-დან 10-მდე ლუწი რიცხვებით, მესამე სტრიქონის ყოველი ელემენტი გახდეს პირველი და მეორე სტრიქონის შესაბამისი სვეტის ელემენტების ჯამი, ხოლო მეოთხე სტრიქონი შეავსეთ 0-ებით,

შეფასების რუბრიკა

 მასივის ორგანიზება მატრიცული სახით ეკრანზე - 2 ქულა

 მასივის ელემენტებით შევსება , გაუშვით პროგრამა შესრულებაზე და პროგრამის კოდთან ერთად გამოაგზავნეთ პროგრამის შესრულების შედეგი - 4 ქულა

```

#include <stdio.h>

#include <stdbool.h>

bool isPrime(int num) {

if (num <= 1) {

return false;

}

for (int i = 2; i \* i <= num; i++) {

if (num % i == 0) {

return false;

}

}

return true;

}

int main() {

int array[4][4];

printf("\n");

int primeCount = 0;

int evenCount = 0;

int k = 1;

for(k; k<10; k++){

if(isPrime(k)){

array[0][primeCount] = k;

primeCount ++;

}

if (k%2==0){

array[1][evenCount] = k;

evenCount++;

}

}

for (int i = 0; i < 4; i++) {

array[2][i] = array[0][i] + array[1][i];

array[3][i] = 0;

}

printf("\n");

for(int i = 0; i < 4; i++) {

for(int j = 0; j < 4; j++ ){

printf("%d\t",array[i][j]);

}

printf("\n");

}

return 0;

}

```

კურსის სახელწოდება: სტრუქტურული დაპროგრამება

ლექტორი: ნონა ოთხოზორია

ამოცანა 1

1. უზრუნველყავით 5 მნიშვნელობის შეტანა კლავიატურიდან. მოითხოვეთ 0-დან 1000-მდე რიცხვის შეტანა.

თითოეული მნიშვნელობის შეტანისას უნდა გამოვიდეს შეტყობინება მოცემულ რიცხვი შეიცავს თუ არა ციფრ „0-ს“ ?

```

#include <stdio.h>

int main() {

int i, num;

for (i = 1; i <= 5; i++) {

printf("Enter a number between 0 and 1000: ");

scanf("%d", &num);

if (num >= 0 && num <= 1000) {

int containsZero = 0;

int temp = num;

while (temp > 0) {

if (temp % 10 == 0) {

containsZero = 1;

break;

}

temp /= 10;

}

if (containsZero) {

printf("The given number %d contains the digit '0'\n", num);

} else {

printf("The given number %d does not contain the digit '0'\n", num);

}

} else {

printf("Please enter a valid number between 0 and 1000.\n");

i--;

}

}

return 0;}

```

შეფასების რუბრიკა

 ციკლის სწორი ორგანიზება - 2 ქულა

 შედარების ორგანიზება - 2 ქულა

 გაუშვით პროგრამა შესრულებაზე და პროგრამის კოდთან ერთად გამოაგზავნეთ პროგრამის შესრულების შედეგი - სურათის სახით.- 1 ქულა

ამოცანა 2

მოცემულია მთელი რიცხვების მასივი. მასივის ელემენტების შეტანა უზრუნველყავით კლავიატურიდან. დასრულდეს მასივის ფორმირება, თუ ერთიდაიგივე რიცხვს შეიტანთ ზედიზედ ორჯერ. დაწერეთ პროგრამა, რომელიც იპოვის კენტინდექსიან წევრებს შორის სიდიდით მეორე მნიშვნელობას.

```

#include <stdio.h>

int main() {

int arr[100];

int n = 0;

while (1){

int num;

printf("Enter element:");

scanf("%d", &num);

if(n > 0 && arr[n-1] == num){

break;

}

arr[n] = num;

n ++ ;

}

int biggest = arr[1];

int secondBiggest = arr[1];

for(int i = 1; i < n; i+=2){

if(arr[i] > biggest){

secondBiggest = biggest;

biggest = arr[i];

}

}

printf("Seccond biggest item from odd index elements : %d", secondBiggest);

return 0;

}

```

შეფასების რუბრიკა

 მასივის შექმნის ორგანიზება - 2 ქულა

 პრობელმის გადაწყვეტა (კენტინდექსიან წევრებს შორის სიდიდით მეორე მნიშვნელობის პოვნა)- 2 ქულა

 გაუშვით პროგრამა შესრულებაზე და პროგრამის კოდთან ერთად გამოაგზავნეთ პროგრამის შესრულების შედეგი - სურათის სახით - 1 ქულა

ამოცანა 3

1. მოცემულია ცხრილი

უნივერსიტეტი ბტუ თსუ სეუ

ბტუ 0 3 5

თსუ 4 0 0

სეუ 3 4 0

შექმენით ორგანზომილებიანი მასივი, რომელიც ასახავს მოცემული უნივერსიტეტის ფეხბურთელთა გუნდის შეხვედრებზე ყვითელი ბარათების რაოდენობას.

გაითვალისწინეთ, რომ მნიშვნელობა , რომელიც განთავსებულია ბტუ (სტრიქონი) - თსუ (სვეტი) უჯრის გადაკვეთაზე მიუთითებს ბტუ-ს ფეხბურთელების მიერ მიღებული ყვითელი ბარათების რაოდენობას მათ ურთერთშეხვედრებში, ხოლო თსუ-ბტუ- უჯრის გადაკვეთაზე - თსუს ფეხბურთელების მიერ მიღებული ყვითელი ბარათების რაოდენობას.

მონაცემები: ბტუ (3) -თსუ ( 4)

თსუ (0)-სეუ (4)

ბტუ(5) - სეუ (3)

დაბეჭდეთ შევსებული მასივი, მატრიცული სახით.

დაადგინეთ თითოეული გუნდის მიერ მიღებული ყვითელი ბარათების რაოდენობა. გამოავლინეთ ყველაზე უხეში გუნდი.

არსებული მატრიცის შედეგების მიხედვით შექმენით სიმბოლური ტიპის მატრიცა. ყველა რიცხვს, რომელიც მეტია 0-ზე, შეუსაბამეთ სიმბოლო ‘1’, ხოლო დანარჩენს სიმბოლო ‘0’, მთავარი დიაგონალის უჯრაზე, რომელიც ყველაზე უხეში გუნდის დასახელებების გადაკვეთას შეესაბამება, ჩასვით სიმბოლო ‘\*’.

```

#include <stdio.h>

int main() {

// Initialize the array with the number of yellow cards

int yellowCards[3][3] = {

{0, 3, 5},

{4, 0, 0},

{3, 4, 0}

};

const char\* universities[] = {"BTU", "TSU", "SEU"};

int numTeams = 3;

int yellowCardCount[numTeams];

// Calculate the total number of yellow cards for each team

for (int i = 0; i < numTeams; i++) {

yellowCardCount[i] = 0;

for (int j = 0; j < numTeams; j++) {

yellowCardCount[i] += yellowCards[i][j];

}

}

// Identify the roughest team

int roughestTeamIndex = 0;

int maxYellowCards = yellowCardCount[0];

for (int i = 1; i < numTeams; i++) {

if (yellowCardCount[i] > maxYellowCards) {

maxYellowCards = yellowCardCount[i];

roughestTeamIndex = i;

}

}

// Create a symbolic type matrix and print the results

printf("Yellow Cards Matrix:\n");

for (int i = 0; i < numTeams; i++) {

for (int j = 0; j < numTeams; j++) {

printf("%d ", yellowCards[i][j]);

}

printf("\n");

}

printf("Yellow Cards Count:\n");

for (int i = 0; i < numTeams; i++) {

printf("Team %s: %d yellow cards\n", universities[i], yellowCardCount[i]);

}

printf("Roughest Team: Team %s\n", universities[roughestTeamIndex] );

// Create a symbolic type matrix

printf("Symbolic Type Matrix:\n");

for (int i = 0; i < numTeams; i++) {

for (int j = 0; j < numTeams; j++) {

if (i == j) {

printf("\* ");

} else if (yellowCards[i][j] > 0) {

printf("1 ");

} else {

printf("0 ");

}

}

printf("\n");

}

return 0;

}

```

შეფასების რუბრიკა

 მასივის ორგანიზება მატრიცული სახით ეკრანზე - 2 ქულა

 მასივის შევსება - 2 ქულა

 ყველაზე უხეში გუნდის გამოვლენა- 3 ქულა

 სიმბოლური მატრიცის შექმნა და გამოტანა ეკრანზე, გაუშვით პროგრამა შესრულებაზე და პროგრამის კოდთან ერთად გამოაგზავნეთ პროგრამის შესრულების შედეგი - სურათის სახით - 3 ქულა.

ვიღაცის ჩაგდებული რაღაცა

ვარიანტი 1

1.

დაწერეთ პროგრამა, რომელიც უზრუნველყოფს კლავიატურიდან რამდენიმე რიცხვის შეტანას.

შეტანილ რიცხვებს შორის, ის რიცხვები, რომლებიც 7-ზე გაყოფისას ნაშთში გვაძლევს 1, 2 ან 5-ს გადაწერეთ მასივში.

მონაცემების შეტანა დასრულდეს, თუ შეტანილი მნიშვნელობა იქნება 0.

#include <stdio.h>

int main() {

int numbers[100]; // An array to store the numbers

int count = 0; // Number of elements in the array

int num;

printf("Enter numbers (enter 0 to end):\n");

do {

scanf("%d", &num);

// Check if the entered number satisfies the condition

if (num != 0 && (num % 7 == 1 || num % 7 == 2 || num % 7 == 5)) {

numbers[count] = num;

count++;

}

} while (num != 0);

if (count > 0) {

printf("Numbers that satisfy the condition:\n");

for (int i = 0; i < count; i++) {

printf("%d ", numbers[i]);

}

printf("\n");

} else {

printf("No numbers found that satisfy the condition.\n");

}

return 0;

}

2.

მოცემულია 10 ელემენტიანი მასივი A[20] {3,4,6,7,12,15,17,18,11,24,15,-9,-20,-8,-10,-16]

დათვალეთ მოცემულ მასივში მარტივი რიცხვების რაოდენობა და გადაწერეთ ეს რიცხვები ახალ bar მასივში.

A მასივში მარტივი რიცხვები ჩაანაცვლეთ 0-ებით.

#include <stdio.h>

#include <stdbool.h>

// Function to check if a number is prime

bool isPrime(int num) {

if (num <= 1) {

return false;

}

if (num <= 3) {

return true;

}

if (num % 2 == 0 || num % 3 == 0) {

return false;

}

for (int i = 5; i \* i <= num; i += 6) {

if (num % i == 0 || num % (i + 2) == 0) {

return false;

}

}

return true;

}

int main() {

int A[20] = {3, 4, 6, 7, 12, 15, 17, 18, 11, 24, 15, -9, -20, -8, -10, -16};

int primeNumbers[20];

int primeCount = 0;

for (int i = 0; i < 20; i++) {

if (isPrime(A[i])) {

primeNumbers[primeCount] = A[i];

primeCount++;

A[i] = 0; // Replace the prime number with 0 in array A

}

}

printf("Prime numbers in array A: ");

for (int i = 0; i < primeCount; i++) {

printf("%d ", primeNumbers[i]);

}

printf("\n");

printf("Updated array A with prime numbers replaced by 0s: ");

for (int i = 0; i < 20; i++) {

printf("%d ", A[i]);

}

printf("\n");

return 0;

}

3.

შექმენით ორგანზომილებიანი მასივი A(4X4). მასივის პირველი სტრიქონი შეავსეთ 1-დან 10-მდე მარტივი რიცხვებით,

მეორე სტრიქონი - 1-დან 10- მდე ლუწი რიცხვებით, მესამე სტრიქონის ყოველი ელემენტი გახდეს პირველი და მეორე

სტრიქონის შესაბამისი სვეტის ელემენტების ჯამი, ხოლო მეოთხე სტრიქონი შეავსეთ 0-ებით,

#include <stdio.h>

int main() {

int A[4][4];

// Fill the first line with simple numbers from 1 to 10

for (int i = 0, num = 1; i < 4; i++) {

A[0][i] = num;

num++;

}

// Fill the second line with even numbers from 2 to 10

for (int i = 0, num = 2; i < 4; i++) {

A[1][i] = num;

num += 2;

}

// Calculate the third line as the sum of corresponding elements from the first and second line

for (int i = 0; i < 4; i++) {

A[2][i] = A[0][i] + A[1][i];

}

// Fill the fourth line with 0s

for (int i = 0; i < 4; i++) {

A[3][i] = 0;

}

// Print the 4x4 array

for (int i = 0; i < 4; i++) {

for (int j = 0; j < 4; j++) {

printf("%2d ", A[i][j]);

}

printf("\n");

}

return 0;

}

ვარიანტი 2

1.

დაწერეთ პროგრამა, რომელიც უზრუნველყოფს კლავიატურიდან რამდენიმე რიცხვის შეტანას, თუ შეტანილი რიცხვი იქნება ორნიშნა,

პროგრამამ უნდა შეაბრუნოს ეს რიცხვი და დაბეჭდოს როგორც საწყისი. ასევე შებრუნებული რიცხვი. მონაცემების შეტანა დასრულდეს,

თუ შეტანილი მნიშვნელობა იქნება 0.

#include <stdio.h>

int reverseNumber(int num) {

int reversed = 0;

while (num > 0) {

int digit = num % 10;

reversed = reversed \* 10 + digit;

num /= 10;

}

return reversed;

}

int main() {

int num;

printf("Enter numbers (enter 0 to end):\n");

do {

scanf("%d", &num);

if (num != 0) {

if (num >= 10 && num <= 99) {

int reversed = reverseNumber(num);

printf("Original: %d, Reversed: %d\n", num, reversed);

} else {

printf("Number is not two digits long: %d\n", num);

}

}

} while (num != 0);

return 0;

}

2.

მოცემულია 10 ელემენტიანი მასივი A[20]={3,4,6,7,12,15,17,18,11,24,15,-9,-20,-8,-10,-16]

რიცხვს ვუწოდოთ „მაღალი“, თუ ის მეტია მის მარჯვნივ და მარცხნივ განლაგებულ რიცხვზე. დაადგინეთ „მაღალი“ რიცხვების რაოდენობა და გადაწერეთ ეს რიცხვები ახალ b\_ar მასივში.

A მასივში „მაღალი“ რიცხვები ჩაანაცვლეთ ()- ებით.

#include <stdio.h>

int main() {

int A[20] = {3, 4, 6, 7, 12, 15, 17, 18, 11, 24, 15, -9, -20, -8, -10, -16};

int B[20];

int highCount = 0;

for (int i = 0; i < 20; i++) {

if (i > 0 && i < 19 && A[i] > A[i - 1] && A[i] > A[i + 1]) {

B[highCount] = A[i];

highCount++;

A[i] = 0; // Replace the high number with 0

}

}

printf("High numbers in array A: ");

for (int i = 0; i < highCount; i++) {

printf("%d ", B[i]);

}

printf("\n");

printf("Updated array A with high numbers replaced by 0s: ");

for (int i = 0; i < 20; i++) {

printf("%d ", A[i]);

}

printf("\n");

return 0;

}

3.

Name:Nino - intermediate -23 point, presentation - 15point, Homework -12 point, FinalProcjet-29point.

Name:Nana- intermediate -20point, presentation - 25point, Homework -10 point, FinalProcjet-28point.

Name:Lika- intermediate -30point, presentation - 16point, Homework -13 point, FinalProcjet-40point.

Name:Maka- intermediate -19 point, presentation - 20point, Homework -15 point, FinalProcjet-29point.

ცხრილის რიცხვითი მონაცემებისათვის შექმენით ორგანზომილებიანი მასივი,

იპოვეთ სიდიდით მეორე მაქსიმალური ქულა მთელ მონაცემებში და დაადგინეთ ვინ მიიღო ეს ქულა და რომელ კომპონენტში?

ეკრანზე გამოვიდეს შეტყობინება ფინალური - 35 მაგ. ნინო

#include <stdio.h>

int main() {

// Create a two-dimensional array to store the data

char names[4][20] = {"Nino", "Nana", "Lika", "Maka"};

int scores[4][4] = {

{23, 15, 12, 29},

{20, 25, 10, 28},

{30, 16, 13, 40},

{19, 20, 15, 29}

};

int highest = -1; // Initialize highest score to a low value

int secondHighest = -1; // Initialize second highest score to a low value

char person[20];

char component[20];

// Iterate through the array to find the second-highest score

for (int i = 0; i < 4; i++) {

for (int j = 0; j < 4; j++) {

if (scores[i][j] > highest) {

secondHighest = highest;

highest = scores[i][j];

strcpy(person, names[i]);

if (j == 0) {

strcpy(component, "Intermediate");

} else if (j == 1) {

strcpy(component, "Presentation");

} else if (j == 2) {

strcpy(component, "Homework");

} else if (j == 3) {

strcpy(component, "Final Project");

}

} else if (scores[i][j] > secondHighest && scores[i][j] < highest) {

secondHighest = scores[i][j];

strcpy(person, names[i]);

if (j == 0) {

strcpy(component, "Intermediate");

} else if (j == 1) {

strcpy(component, "Presentation");

} else if (j == 2) {

strcpy(component, "Homework");

} else if (j == 3) {

strcpy(component, "Final Project");

}

}

}

}

// Print the second highest score, person, and component

printf("Second highest score: %d, %s scored in %s\n", secondHighest, person, component);

return 0;

}