Znenie zadania:

Implementujte MAXMIN algoritmus podľa uvedeného pseudokódu (alebo prednášok).

Opis riešenia:

Funkcia maxmin hľadá minimálny a maximálny prvok postupnosti s využitím metódy DnC. Na začiatku funkcie sa vypíše pole, s ktorým funkcia pracuje. Ak počet prvkov je rovný jedna, daný prvok je minimom aj maximom. Ak sú prvky 2 iba sa jednoducho porovnajú a do premennej max sa uloží väčší z prvkov a do premennej min menší z prvkov. Ak je prvkov viac ako 2, volá sa rekurzívne volanie funkcie, kde sa najprv osobitne vyhodnotí prvá a druhá polovica poľa. Vypočítané minimá a maximá sa porovnajú a uložia do príslušných premenných. Na konci funkcie sa do štruktúry dvoch integerov uložia minimum a maximum a táto štruktúra sa odovzdáva ako návratová hodnota funkcie.

Zdrojový kód:

funkcia maxmin(int n,int *r, int index):

```
retVal  maxmin(int n,int *r, int index){
    int i, max, min;
    retVal  return_maxmin, left_maxmin, right_maxmin;

    printf("\nHladam max a min s %d prvkoveho pola: ",n);
    for (i=index; i<index+n; i++) printf("%2d; ",r[i]);
    printf(".\n");

    if(n==1){
        max=r[index];
        min=r[index];
    }

    else if(n==2){
        if(r[0]>r[1]){
            max=r[index];
            min=r[index+1];
        }else{
            max=r[index+1];
            min=r[index];
        }

    }else{
        printf("rekurzia");
        left_maxmin=maxmin(n/2,r,index);
        right_maxmin=maxmin((n-(n/2)),r,index+n/2);

        max = (left_maxmin.max<right_maxmin.max) ? right_maxmin.max :
               left_maxmin.max;
        min = (left_maxmin.min<right_maxmin.min) ? left_maxmin.min :
               right_maxmin.min;

    }


    return_maxmin.max = max;
    return_maxmin.min = min;
    return return_maxmin;
}
```

Znenie zadania:

Implementujte CHAIN_MATRIX_MULTIPLICATION algoritmus podľa uvedeného pseudokódu (alebo prednášok).

Opis riešenia:

Funkcia hľadá minimálnu cenu násobenia matíc. Najprv v tabuľke reprezentovanej dvojrozmerným poľom vynuluje diagonálu – násobenie matice sebou samou v tomto prípade nenastane. Potom nasleduje cyklus v ktorom sa vyhodnotí cena násobenia. Prechádza sa postupne tabuľkou, pričom sa vypočítavajú všetky možné možnosti, ale do poľa tabuľky sa uloží iba najmenšia hodnota. Cyklus sa opakuje až kým nie sú zohľadnené všetky možnosti. Nakoniec sa už iba vypíše minimálna cena násobenia.

Zdrojový kód:

funkcia Chain_Matrix_Multiplication(int n, int *r):

```
void Chain_Matrix_Multiplication(int n, int *r){
    int i,j,l,k,min;
    int m[n][n+1];

    for(i=1;i<=n;i++){
        m[i][i]=0;
    }

    for(l=1;l<=n-1;l++){
        for(i=1;i<=n-l;i++){
            printf("\n");
            j=i+l;
            m[i][j]=m[i][i]+m[i+1][j]+r[i-1]*r[i]*r[j];
            for(k = i; k<j; k++){
                min=m[i][k]+m[k+1][j]+r[i-1]*r[k]*r[j];
                if (m[i][j]>min) m[i][j]=min;

                printf("min (%d,%d) = %d \t m[%d][%d] =
%d\n",i,j,min,i,j,m[i][j]);

            }
        }
    }

    printf("\n\n Minimalna cena nasobenia matic ");
    for(j=0;j<=n;j++){ printf("%d; ",r[j]); }
    printf("je: %d\n",m[1][n]);


}
```

Znenie zadania:

Implementujte algoritmus "Coin Changing Problem" podľa nižšie uvedeného pseudokódu (Greedy metóda). Uvažujte riešenia pre rôzne vstupné sumy, ak dostupné sú mince s nominálnymi hodnotami: 100,25,20,5,1.

Opis riešenia:

Funkcia má z určených hodnôt poskladať žiadanú hodnotu ktorá je udaná ako parameter. To sa deje v jednom cykle kde sa postupne skúšajú všetky nominálne hodnoty, kým súčet už odskúšaných a skúšanej je menší alebo rovný požadovanej hodnote. Nájdená hodnota sa vypíše na obrazovku a opakuje sa cyklus. Ak aj najmenšia z možných hodnôt je väčšia ako požadovaná hodnota, vypíše sa chybové hlásenie a vykonávanie cyklu sa ukončí.

Zdrojový kód:

funkcia make_change(int n):

```
void make_change (int n){
    int c[] = {100, 25, 20, 5, 1};
    int Sum=0;
    int i=0;

    while (Sum != n){
        if (c[sizeof(c)]>n){printf("No Solution"); break;}

        if (Sum+c[i]>n){
            for (i = 0; i<=sizeof(c)-1; i++){
                if (Sum + c[i] <= n)  break;
            }
        }
        printf("%d ",c[i]);
        Sum += c[i];
    }
}
```