Znenie zadania:

Doplňte do tela funkcie RadixSortQueue kód realizujúci samotné triednie. Postupne zobrazujte činnosť algoritmu pri každom prechode hlavnou slučkou.

Opis riešenia:

Pre doplnenie výpisu bolo potrebné iba vhodne umiestniť funkcie printf(). Výpis poľa sa vykonáva pomocou cyklu v ktorom sa opakuje výpis hodnoty prvku.

Na doplnenie RadixSortQueue bolo potrené iba upraviť funkcie aby nepracovali s poľom ale s údajovou štruktúrou front. Opäť sa volajú rovnaké cykly, v ktorých sa najprv prechádzajú jednotlivé pozície čísel a na základe ich hodnoty sa ukladajú do príslušného zoznamu.

Zdrojový kód:

funkcia RadixSort(char* A[ ], int n, int k):

```
void RadixSort(char* A[],int n,int k){
     char* B[10][n];
     int pocB[10];
     int x,y;
     int i,j;
     for(i=0;i<10;i++)pocB[i]=0;
     for(j=k-1;j>=0;j--){
         printf("\n --- j = %d ---",j);

         for(i=0;i<n;i++){
             x=A[i][j]-'0';
             B[x][pocB[x]++]=A[i];

             }
          for(y=0,x=0;x<10;x++){
              printf("\nB[%d]: ",x);
              for(i=0;i<pocB[x];i++){
                  A[y++]=B[x][i];
                  printf(" %s",B[x][i]);
              }
              pocB[x]=0;
              }

         printf("\nQ:");
         for(i=0; i<n; i++){
             printf(" %s",A[i]);
         }
         printf("\n");
     }
     printf("\n");
}
```

funkcia RadixSortQueue(char* A[ ], int n, int k):

```
void RadixSortQueue(char* A[],int n,int k){
     int x,y;
     int i,j;
     TQUEUE* q;
     TQUEUE* B[10];
     TNODE *temp;
     q=CreateQueue();
     for(i=0;i<10;i++)B[i]=CreateQueue();
```

```
    for(i=0;i<n;i++){
        temp=CreateNode(A[i]);
        Enqueue(q,temp);
     }
       for(i=k-1;i>=0;i--){
           printf("\n --- j = %d ---\n", i);
           while(!IsEmpty(q)){
               temp = FrontAndDequeue(q);
               x = *(temp->value+i)-'0';
               Enqueue(B[x],temp);
           }
           for(x=0;x<10;x++){
               printf("B[%d]: ", x);
               PrintQueue(B[x]);
               while(!IsEmpty(B[x])){
                   Enqueue(q, FrontAndDequeue(B[x]));
               }
           }
           printf("\nQ: ");
           PrintQueue(q);
           printf("\n");
     }



    i=0;
    while(!IsEmpty(q)){
        temp=FrontAndDequeue(q);
        A[i++]=temp->value;
    }
}
```