Application for CNC machine

**Martin Maasik**

Table of contents

[Table of contents: 2](#_Toc45634434)

[Introduction 3](#_Toc45634435)

[Summary of the solution 3](#_Toc45634436)

[Disc: 4](#_Toc45634437)

[Segment: 4](#_Toc45634438)

[Required application 5](#_Toc45634439)

[Data to be entered by the user: 5](#_Toc45634440)

[Drilling cross section per line: 6](#_Toc45634441)

[Drilling cross section by rows: 6](#_Toc45634442)

[WORKING PRINCIPLE OF THE DISK PROGRAM 7](#_Toc45634443)

[WORKING PRINCIPLE OF THE SECTOR PROGRAM 7](#_Toc45634444)

[Actual solution 8](#_Toc45634445)

[Win Forms UI 8](#_Toc45634446)

[Buttons 9](#_Toc45634447)

[Create program button: 9](#_Toc45634448)

[Save (txt) button: 11](#_Toc45634449)

[**Copy** button: 11](#_Toc45634450)

[Save (csv) button 12](#_Toc45634451)

[Load database button: 12](#_Toc45634452)

[Load ID button: 13](#_Toc45634453)

[Database 15](#_Toc45634454)

[Displaying database on webpage 16](#_Toc45634455)

[Index.html 16](#_Toc45634456)

[Select.php 16](#_Toc45634457)

[Insert.php 16](#_Toc45634458)

[Edit.php 16](#_Toc45634459)

[Delete.php 16](#_Toc45634460)

# Introduction

The choice of the topic of current application is a need for automation of some of production processes. Current solution is developed for certain CNC machine to generate machine programs from input data.

# Summary of the solution

Current CNC device produces side brushes for street sweepers and the application calculates the holes positions for different products.

Drilling program consisting of sets or ’Hole pattern’. There are five different variables in one set: Set; Diameter; Start angle; Angle betw. holes; No of holes;

The program may consist of a minimum of one set, a maximum of at least 40 sets.

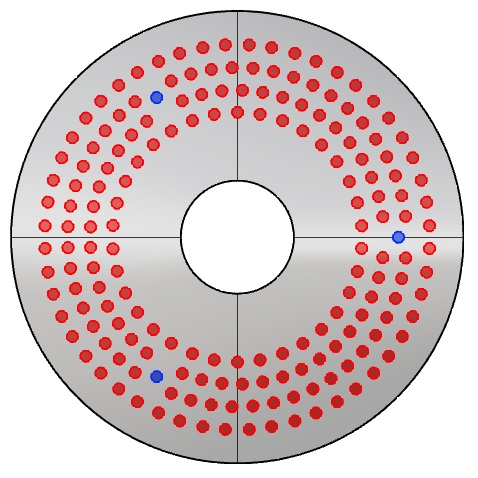
(a disc with only one row of holes and no mounting bolts will only need one set)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | **Set:** | |  | | --- | |  | | | **Diameter:** | |  | | --- | |  | | | **Start angle:** | |  | | --- | |  | | | **Angle betw.holes:** | |  | | --- | |  | | | **No. of holes:** | |  | | --- | |  | | |

*Data fields on CNC*

* **Set** – set number. Starts with zero and increases by one unit.
* **Diameter** – diameter for current row of holes(set).
* **Start angle** – start angle for the current set(0-360 degrees).
* **Angle betw.holes** – angle between the holes in degrees. In current application one hole represents a pair of holes. The full circle is divided into 360 degrees and one degree is divided by 100.
* **No. of holes** – the number of holes in a given set.

## Disc:

    The disc shown in this figure has four rows of holes, two of which are divided into three sectors by three fixing bolts(blue circles). Thus, a total of 8 sets (0-7) are required for this product.

The diameter of the rows to be drilled will be calculated starting from the outer row (the diameter of the outer row is the default disc diameter – 60 mm and the value between the rows is 40 mm).

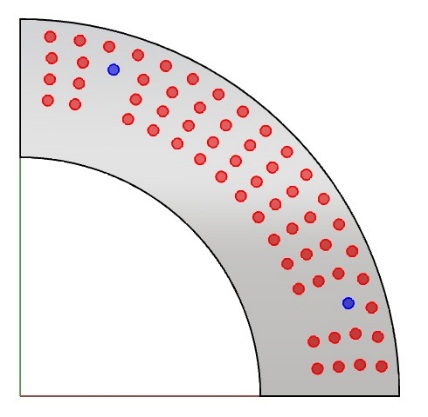
Drilling starts from the inner row and moves outwards.

The drilling of the inner row starts from 0 degree, moving up to 360 degrees and if the disc has mounting holes, one of them will be on the 0 degree angle during production process. Every second row of holes starts at 360 degree, moving back to 0 and so on. So rows 2, 4, 6, 8 ets. hole pairs are thus denoted by a (-) negative sign.

*blue dots – fixing holes*

*red dots – holes to be drilled*

## Segment:

In case of a segment, all rows smaller than the fixing hole row are divided into sectors. The program starting point for segment will never be 0 degrees.

Drilling will start from the inner row clockwise, moving alternately clockwise / counterclockwise outwards.

*blue dots – fixing holes*

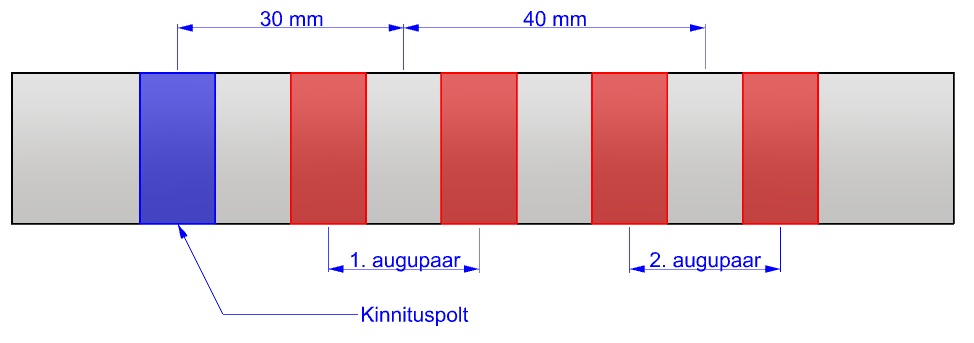
*red dots – holes to be drilled*

# Required application

## Data to be entered by the user:

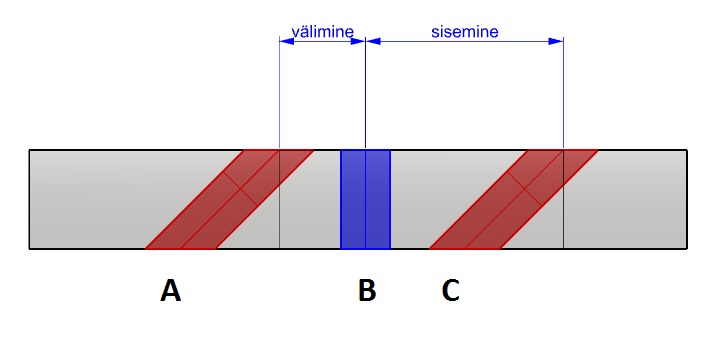
|  |  |
| --- | --- |
| **Ketta läbimõõt:** | |
|  | Diameter of the disc to be drilled in millimeters. |
| **Ridade arv:** | |
|  | Number of rows. At least 1 row is needed. The number of rows and the diameter of disc are minimum data on the basis of which one set can be calculated. |
| **Kinnituspoltide arv:** | |
|  | If the disc has mounting bolt holes, the user will be asked for their amount. |
| **Kinnituspoltide rea läbimõõt:** | |
|  | Diagonal of the row of mounting bolt holes. |
| **Segmendi algus:** | |
|  | Starting angle for segment(used only on sectors). |
| **Segmendi lõpp:** | |
|  | End angle for segment(used only on sectors). |
| **Esimene kinnituspolt:** | |
|  | Angle of first mounting bolt(used only on sectors). |
| **Teine kinnituspolt:** | |
|  | Angle of the second mounting bolt(used only on sectors). |
| **Augupaaride vahe:** | |
|  | Distance between hole pairs. The default setting is 40mm. |
| **Puuritava ketta ja välimise aukuderea vahe:** | |
|  | Difference between disc and outer row diameter. The default setting is 60mm. |
| **Puuritavate auguridade vahe:** | |
|  | Distance between rows. The default setting is 40mm. |
| **Puuritava augurea algus kinnituspoldist:** | |
|  | Set start angle from mounting bolt hole. The default setting is 30mm. |
| **Kinnituspoldirea ja sellest väiksema puuritava aukudera vahe:** | |
|  | Distance between mounting bolts hole row and smaller drilling holes row. The default value is 64mm. |
| **Kinnituspoltiderea ja sellest suurema puuritava aukuderea vahe:** | |
|  | Distance between mounting bolts hole row and bigger drilling holes row. The default value is 35mm. |

# Drilling cross section per line:



1. **Optimal distance between mounting bolt hole and drilling hole pair is 30mm and distance between drilling hole pairs is 40mm.**

# Drilling cross section by rows:



A and C(red) – holes are always drilled outwards from the center of the disc in an angle(25-45 degrees).

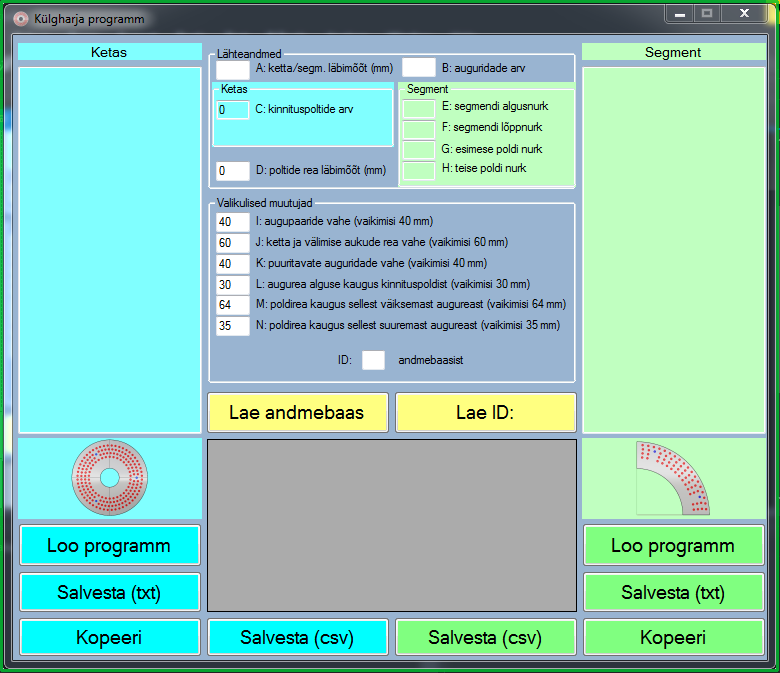
B – mounting bolt holes(blue) are always drilled at 90-degree angle. In order to prevent fastening bolt hole from being left in front of the holes to be drilled, it is necessary to „cut into sectors“ rows that get too close to it.

|  |  |
| --- | --- |
| WORKING PRINCIPLE OF THE DISK PROGRAM | |
|  | *The disc drilling program starts clockwise with a full circle or cector, depending on wheter the mounting bolts are in front of the row of holes or not. The next line will be drilled counterclockwise. The program ends when all conditions are met – the sets of all rows have been calculated.*  *(A full circle gives only one set)* |

|  |  |
| --- | --- |
| WORKING PRINCIPLE OF THE SECTOR PROGRAM | |
|  | *The course of the segment drilling program lasts differently, but it also stops at the point where all predetermined conditions are met – sets of all rows have been created.* |

# Actual solution

## Win Forms UI



In the Win Forms user environmet, two types of programs can be created: for disks and for segments(sectors). Items with a light blue background apply only to the disk, and items with a light green background apply to the sector. Seperate text output fields is also used to display results.

A total of 14 data entry fields (A-N) are used to create the programs. The first four of these (A-D) are most important and basic data provided for new products.

Only numeric values are allowed in text fields:

private void AtxtBox\_KeyPress(object sender, KeyPressEventArgs e)

{

// only numeric input values allowed

char ch = e.KeyChar;

if ((!Char.IsDigit(ch) && ch != 8) && ch > '.')

{

e.Handled = true;

}

}

# Buttons



## Create program button:

* Firstly output field will be cleared.

ketasTxtBox.Clear();

* Tekstiväljale kuvatakse kolm rida toote info kohta.

ketasTxtBox.Text += "Toote kood: \r\nKiu pikkus: \r\nKaldenurk: \r\n";

* It is checked wheter the values have been entered in text fields A and B. Otherwise, an error message is displayed.

if (string.IsNullOrWhiteSpace(AtxtBox.Text) || tring.IsNullOrWhiteSpace(BtxtBox.Text))

{

MessageBox.Show("Täitke vähemalt A: ja B: väljad!");

return;  
}

* It is checked wheter only one of the C and D fields has a numeric value added, as these two fields can not be used simultaneously.

else if (string.IsNullOrWhiteSpace(CtxtBox.Text) && !string.IsNullOrWhiteSpace(DtxtBox.Text) || !string.IsNullOrWhiteSpace(CtxtBox.Text) && string.IsNullOrWhiteSpace(DtxtBox.Text))

{

MessageBox.Show("C: ja D: välja saab vaid üheaegselt kasutada!");

return;

}

* Check that none of the fields I-N are empty.

else if (string.IsNullOrWhiteSpace(ItxtBox.Text))

{

MessageBox.Show("Tekstiväli I on tühi!");

return;  
 }

* Once the previous checks have been passed, the further calculation process is selected according to the source data.

try

{

// obtaining values for key variables from text fields

kpArv = 0;

int.TryParse(CtxtBox.Text, out kpArv);

kpDiam = 0;

int.TryParse(DtxtBox.Text, out kpDiam);

int kettaLabimoot = int.Parse(AtxtBox.Text);

int auguridadeArv = int.Parse(BtxtBox.Text);

kpArv = int.Parse(CtxtBox.Text);

kpDiam = int.Parse(DtxtBox.Text);

// obtaining values for optional variables

aVahe = int.Parse(ItxtBox.Text);

vRida = int.Parse(JtxtBox.Text);

rVahe = int.Parse(KtxtBox.Text);

kaugusKP = int.Parse(LtxtBox.Text);

poltidestV = int.Parse(MtxtBox.Text);

poltidestS = int.Parse(NtxtBox.Text);

int set = 0; // number of the first set

int diam = kettaLabimoot - vRida - (rVahe \* (auguridadeArv - 1)); // diameter of the first set

int ak = 0; // total number of all hole pairs

if (diam > kpDiam + poltidestS || diam < kpDiam - poltidestV) // clockwise whole row

{

// will be moved clockwise whole row

pptr(ketasTxtBox, ak, kettaLabimoot, auguridadeArv, kpArv, kpDiam, aVahe, vRida, rVahe, kaugusKP, poltidestV, poltidestS, set, diam, start, dnoh, noh, angl);

}

else // clockwise sector

{

// will be moved clockwise sector

ppsk(ketasTxtBox, ak, kettaLabimoot, auguridadeArv, kpArv, kpDiam, aVahe, vRida, rVahe, kaugusKP, poltidestV, poltidestS, set, diam, start, dnoh, noh, angl);

}

}

catch

{

MessageBox.Show("ERROR");  
 }

Two similar buttons for different programs.

Both buttons are tested and work as intended. The generated programs are usable in current CNC machine.



## Save (txt) button:

* Initiates a save dialog during which the user can save the contents of the program text field in .txt format.

There are two similar buttons for different programs.

Both buttons tested and working according their purpose.

*Click event:*

// Saves program in a text file

private void ketasSaveBtn\_Click(object sender, EventArgs e)

{

SaveFileDialog saveFileDialog1 = new SaveFileDialog();

saveFileDialog1.Filter = "Text Files | \*.txt";

saveFileDialog1.DefaultExt = "txt";

if (saveFileDialog1.ShowDialog() == DialogResult.OK)

{

using (Stream s = File.Open(saveFileDialog1.FileName, FileMode.CreateNew))

using (StreamWriter sw = new StreamWriter(s))

{

sw.Write(ketasTxtBox.Text);

}

}

}



## **Copy** button:

* Checks if anything is entered in the program text field.
* If so, the contents of the text field are copied to the cache so that it can be pasted if desired.

There are two similar buttons for different programs.

Both buttons tested and working according their purpose.

*Click event:*

// Copies content of disk program text field

private void ketasCopyBtn\_Click(object sender, EventArgs e)

{

// Checks if anything is entered on to the text field

if (!string.IsNullOrWhiteSpace(ketasTxtBox.Text))

{

Clipboard.SetText(ketasTxtBox.Text);

}

else

{

return;

}

}

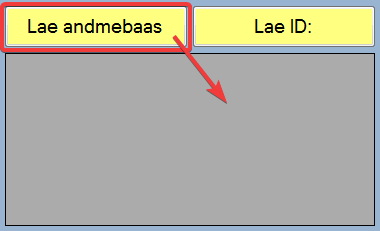


Save (csv) button:

Currently reserved. Intended to be able to save the calculated program in .csv format, in whitch the programs of the CNC machine in question are stored. It is currently not possible to use this feature because the device manufacturer does not give description for .csv format values(all automated entries).

There are two similar buttons for different programs.

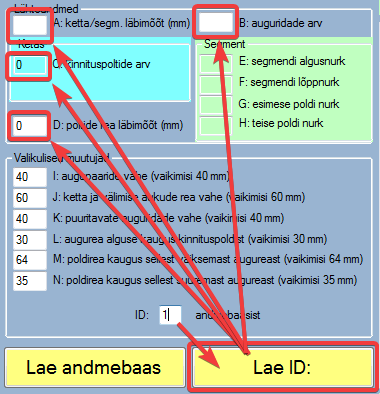
Reserved for possible update.



## Load database button:

* Loads the product source data in the database into the UI gridview field.

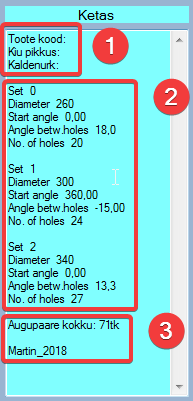
There is only one button given thanks to the common database.



## Load ID button:

* Displays values from the database for the first four data fields (A-D) according to the id number in the ID: text input field

There is one button that applies to both programs



A text field that displays the created program. The information displayed in the field consists of three parts:

1. Automatically generated rows to which the folder can add entries if desired (Product code; Fiber length; Tilt angle).
2. The sample program consists of three sets, each with 5 values (Set; Diameter; Start angle; Angle betw. Holes; No. of holes).
3. At the end of the program, the total number of hole pairs is also displayed (in this example 71). This is necessary when calculating the price of the product.

# Database

In addition that this application calculates the result of the data entered on user input, it also has the possibility to connect to the database containing the source data of the products.

Sample database created for the program with five variables: id; Name(product name) and four variables / source data from which the programs are generated

**CREATE TABLE `harjad` (  
 `id` int(11) NOT NULL,  
 `Nimi` varchar(15) COLLATE utf8\_estonian\_ci NOT NULL,  
 `Aklm` int(3) NOT NULL,  
 `Brda` int(1) NOT NULL,  
 `Ckia` int(2) NOT NULL,  
 `Dkil` int(3) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_estonian\_ci;**

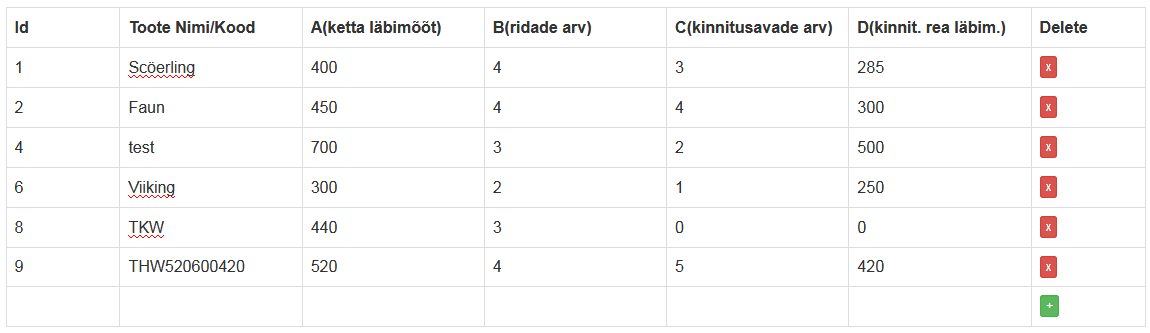
Some entries added to the table:

**INSERT INTO `harjad` (`id`, `Nimi`, `Aklm`, `Brda`, `Ckia`, `Dkil`) VALUES  
(1, 'ScÃ¶erling', 400, 4, 3, 285),  
(2, 'Faun', 450, 4, 4, 300),  
(4, 'test', 700, 3, 2, 500),  
(6, 'Viiking', 300, 2, 1, 250),  
(8, 'TKW', 440, 3, 0, 0),  
(9, 'THW520600420', 520, 4, 5, 420);**

# Displaying database on webpage

## Index.html

A web page (index.html) is required to view, enter, modify and delete the database in real time. On this page, the database information is displayed in the form of a table as follows:



## Select.php

Connection to the database is created in select.php:

**$connect = mysqli\_connect("localhost", "root", "", "KH");**

And quiery to the database:

**$sql = "SELECT \* FROM harjad ORDER BY id";**

## Insert.php

New entry can be added in insert.php’s:

**$sql = "INSERT INTO harjad(Nimi, Aklm, Brda, Ckia, Dkil) VALUES('".$\_POST["Nimi"]."', '".$\_POST["Aklm"]."', '".$\_POST["Brda"]."', '".$\_POST["Ckia"]."', '".$\_POST["Dkil"]."')";**

## Edit.php

Existing data is modified in real time when user clicks on a cell in the table displayed on the index.html page and enters a new value. This change is made using edit.php:

**$id = $\_POST["id"];   
 $text = $\_POST["text"];   
 $column\_name = $\_POST["column\_name"];   
 $sql = "UPDATE harjad SET ".$column\_name."='".$text."' WHERE id='".$id."'";**

## Delete.php

Entries are deleted by clicking on a red x in the ’Delete’ cell of the table. Then delete.php’s query the database:

**$sql = "DELETE FROM harjad WHERE id = '".$\_POST["id"]."'";**