# Appendix 1: Java code

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137  138  139  140  141  142  143  144  145  146  147  148  149  150  151  152  153  154  155  156  157  158  159  160  161  162  163  164  165  166  167  168  169  170  171  172  173  174  175  176  177  178  179  180  181  182  183  184  185  186  187  188  189  190  191  192  193  194  195  196  197  198  199  200  201  202  203  204  205  206  207  208  209  210  211  212  213  214  215  216  217  218  219  220  221  222  223  224  225  226  227  228  229  230  231  232  233  234  235  236  237  238  239  240  241  242  243  244  245  246  247  248  249  250  251  252  253  254  255  256  257  258  259  260  261  262  263  264  265  266  267  268  269  270  271  272  273  274  275  276  277  278  279  280  281  282  283  284  285  286  287  288  289  290  291  292  293  294  295  296  297  298  299  300  301  302  303  304  305  306  307  308  309  310  311  312  313  314  315  316  317  318  319  320  321  322  323  324  325  326  327  328  329  330  331  332  333  334  335  336  337  338  339  340  341  342  343  344  345  346  347  348  349  350  351  352  353  354  355  356  357  358  359  360  361  362  363  364  365  366  367  368  369  370  371  372  373  374  375  376  377  378  379  380  381  382  383  384  385  386  387  388  389  390  391  392  393  394  395  396  397  398  399  400  401  402  403  404  405  406  407  408  409  410  411  412  413  414  415  416  417  418  419  420  421  422  423  424  425  426  427  428  429  430  431  432  433  434  435  436  437  438  439  440  441  442  443  444  445  446  447  448  449  450  451  452  453  454  455  456  457  458  459  460  461  462  463  464  465  466  467  468  469  470  471  472  473  474  475  476  477  478  479  480  481  482  483  484  485  486  487  488  489  490  491  492  493  494  495  496  497  498  499  500  501  502  503  504  505  506  507  508  509  510  511  512  513  514  515  516  517  518  519  520  521  522  523  524  525  526  527  528  529  530 | /\*\*   \* Sort of a simulation of the PP2 program controlling the Fischer Technik in order to sort black and white discs.   \* @team Group 16   \* @author Maarten Keet   \* @author stefan van den Berg   \* @author Rolf Verschuuren   \* @since 13/3/2015   \*/      class SoftwareDesign {        //\*\*@DATA\*\*    //outputs   int[] $outputs=new int[8];      //\*\*@CODE\*\*    //inputs    Boolean $startStop,$abort,$push,$position,$colour;    int $timer;    //variables    int $state=0;    int $sleep=0;    int $location;    int $counter;    int $engines;      //constants    final int $timeMotorDown=-1;    final int $belt=-1;    final int $sort=-1;    final int $timerSort=-1;    final int $lensLampPosition=0,$lensLampSorter=1,$hbridge1=2,$hbridge0=3, $conveyorBelt=4,$feederEngine=5,$display=6,$ledStateIndicator=7;            void initial() {      timerManage($outputs);      $push=buttonPressed(5);      if($push==true){          $outputs[$hbridge0]=0;          $outputs[$hbridge1]=9;          $state = 1;          display($state,"leds2","");          calibrateSorter();        }      initial();    }    void calibrateSorter(){        timerManage($outputs);        if($sleep==$timeMotorDown\*1000){           $outputs[$hbridge1]=9;           $state=2;           display($state,"leds","");           resting();           $sleep=0;        }        $sleep++;        calibrateSorter();    }      void resting(){        timerManage($outputs);        $startStop=buttonPressed(0);        if($startStop==true){            $outputs[$lensLampPosition]=12;            $outputs[$lensLampSorter]=12;            $outputs[$conveyorBelt]=9;            $outputs[$feederEngine]=5;            setTimer(2+$belt);                $state=3;            display($state,"leds2","");            running();        }        resting();    }      void running(){        timerManage($outputs);        $position=buttonPressed(7);        $startStop=buttonPressed(0);        if($startStop=true){            $outputs[$feederEngine]=0;            setTimer($belt);            runningTimer();        }        if($position=true){            setTimer(2+$belt);              $state=4;            display($state,"leds2","");            runningWait();        }        running();    }      void runningWait(){        timerManage($outputs);        $position=buttonPressed(7);        $colour=buttonPressed(6);        $startStop=buttonPressed(0);        if($startStop=true){            $outputs[$feederEngine]=0;            setTimer($belt);            runningTimer();        }        if($position){             setTimer(2+$belt);              $state=5;            display($state,"leds2","");            runningTimerReset();        }        if($colour){            $outputs[$hbridge0]=9;              setTimer($sort);              $state=6;            display($state,"leds2","");            motorUp();        }        runningWait();    }      void runningTimerReset(){        timerManage($outputs);        runningWait();    }      void motorUp(){        timerManage($outputs);        $push=buttonPressed(7);        $startStop=buttonPressed(0);        if($startStop=true){            $outputs[$feederEngine]=0;            setTimer($belt);            motorUpTimer();        }        if($push=true){            $outputs[$hbridge0]=0;            $state=7;            display($state,"leds2","");            whiteWait();        }    }     void whiteWait(){        timerManage($outputs);        if($sleep==$timerSort\*1000){       $outputs[$hbridge1]=9;        $state=8;        display($state,"leds2","");        motorDown();        $sleep=0;        }        $startStop=buttonPressed(0);        if($startStop=true){            $outputs[$feederEngine]=0;            setTimer($belt);            whiteWaitTimer();        }        $sleep++;        whiteWait();    }     void motorDown(){       timerManage($outputs);       if($sleep==$timeMotorDown\*1000){           $outputs[$hbridge1]=0;           $state=9;           $sleep=0;           display($state,"leds2","");           runningWait();       }       $startStop=buttonPressed(0);       if($startStop=true){            $outputs[$feederEngine]=0;            setTimer($belt);            motorDownTimer();        }       $sleep++;       motorDown();     }     void runningTimer(){       timerManage($outputs);       runningStop();   }     void motorUpTimer(){       timerManage($outputs);       motorUpStop();   }     void whiteWaitTimer(){       timerManage($outputs);       whiteWaitStop();   }     void motorDownTimer(){       timerManage($outputs);       motorDownStop();   }       void runningStop(){       timerManage($outputs);       $colour=buttonPressed(6);       if($colour==true){           $outputs[$hbridge0]=9;           $state=10;           display($state,"leds2","");           motorUpStop();       }       runningStop();   }     void motorUpStop(){       timerManage($outputs);       $push=buttonPressed(5);       if($push==true){           $outputs[$hbridge0]=0;            $state=11;            display($state,"leds2","");       }       motorUpStop();   }     void whiteWaitStop(){        timerManage($outputs);        if($sleep==$timerSort\*1000){        $outputs[$hbridge1]=9;        $state=12;        display($state,"leds2","");        motorDown();        $sleep=0;        }          $sleep++;        whiteWait();   }     void motorDownStop(){       timerManage($outputs);       if($sleep==$timeMotorDown\*1000){           $outputs[$hbridge1]=0;           $state=9;           $sleep=0;           display($state,"leds2","");           runningWait();       }       $sleep++;       motorDown();   }     void timerInterrupt(){       timerManage($outputs);       $outputs[$hbridge0]=1;       $outputs[$hbridge1]=0;       $outputs[$lensLampPosition]=0;       $outputs[$lensLampSorter]=0;       $outputs[$ledStateIndicator]=0;       $outputs[$display]=0;       $outputs[$conveyorBelt]=0;       $outputs[$feederEngine]=0;       initial();     }     void abort(){       timerManage($outputs);       $outputs[$hbridge0]=0;       $outputs[$hbridge1]=0;       $outputs[$lensLampPosition]=0;       $outputs[$lensLampSorter]=0;       $outputs[$ledStateIndicator]=0;       $outputs[$display]=0;       $outputs[$conveyorBelt]=0;       $outputs[$feederEngine]=0;       aborted();     }     void aborted(){       timerManage($outputs);       $startStop=buttonPressed(0);       if($startStop=true){           $outputs[$hbridge0]=1;           $state=0;           display($state,"leds2","");           initial();       }       aborted();     }     void timerManage(int[] $outputs){    $location = $location % 7;    $counter = $counter % 12;      if($counter < $outputs[$location]){     $engines = $engines + pow(2, $location);    }      if($location >= 7){     display($engines, "leds","");     $engines = 0;     return;    }      $location++;    $counter++;    timerManage($outputs);    return;   }          public static void main( String args[] ) {      new SoftwareDesign().initial();    }        //public voids for autocorrect  public void sleep(int $seconds)  {}    /\*\*   \* Store a value in the ram.   \*   \* Example: \_storeRam($location,$value)   \*   \* @param variable $location The location to store the value in the ram   \* @param variable $value    The value to store   \*   \* @return void   \*/  public void \_storeRam(int $location,int $value)  {  }    /\*\*   \* Get a value from the ram.   \*   \* Example: $value=\_getRam($location)   \*   \* @param variable $location The location where the value is stored   \*   \* @return void   \*/  public void \_getRam( int $location)  {  }    /\*\*   \* Display something on either the display or the leds   \*   \* Possible values for $onwhat:   \* leds: the leds at the top   \* leds2: the leds to the right   \* display: the display   \* Example:   \* display($value, 'display',000100)   \* This will display $value in the middle of the display   \*   \* @param variable $what     what to display   \* @param variable $onWhat   on what to display   \* @param string   $location Where to show the value when using the display,   \*                           defaults to the right position   \*   \* @return void   \*/  public void display(int $what, String $onWhat, String $location)  {  }    /\*\*   \* Take the mod of a number   \*   \* Example: modulo($variable,2)   \* This will return the mod 2 of $variable   \*   \* @param variable $variable variable to modulo over   \* @param int      $what     modulo what   \*   \* @return void   \*/  public void modulo(int $variable, int $what)  {  }    /\*\*   \* Get button or analog input   \*   \* When you just want hte input of 1 button, use buttonPressed instead   \* Example: getInput($variable,'analog')   \* This will put the value of the analog into $variable   \*   \* @param variable $writeTo Variable to write the input to   \* @param string   $type    Type of input, possible values are: buttons, analog   \*   \* @return void   \*/  public void getInput(int $writeTo, int $type)  {  }    /\*\*   \* Check if a button is pressed   \*   \* Puts the result into R5   \* Example:buttonPressed($location);   \* if (R5 == 1) {}   \*   \* @param variable $button Which button to check   \*   \* @return boolean   \*/  public boolean buttonPressed(int $button)  {      return true;  }    /\*\*   \* Install the countdown   \*   \* Do not forget to add returnt at the end of the interrupt public void   \* Example: installCountdown('timerInterrupt')   \* This will install the countdown.   \* In this example when the timer interrupt triggers,   \* the public void timerInterrupt is ran.   \*   \* @param string int $public voidName The name of the public void where the timer should go to   \*   \* @return void   \*/  public void installCountdown(String $public)  {  }    /\*\*   \*Start the countdown.   \*   \* @return void   \*/  public void startCountdown()  {  }    /\*\*   \*Push a variable to the stack   \*   \* @param string $variable the variable to push to the stack   \*   \* @return void   \*/  public void pushStack(int $variable)  {  }    /\*\*   \*Pull a variable from the stack   \*   \* @param string $variable the variable where the pulled variable is put into   \*   \* @return void   \*/  public void pullStack(int $variable)  {  }    /\*\*   \* Set the timer interrupt to a value.   \*   \* It will first reset the timer to 0.   \* Example: setTimer(10)   \* This will interrupt the program after 10 timer ticks   \*   \* @param string $timer how long the timer should wait, in timer ticks   \*   \* @return void   \*/  public void setTimer(int $timer)  {  }      /\*\*   \* Get data   \*   \* Use offset 0 when it is just a single value.   \* Example: $data=\_getData('data',1)   \* This will put the value of the data segment "data" at position 1, into $data.   \*   \* @param string $location The location where the variable is stored   \* @param int    $offset   The offset of the location   \*   \* @return mixed The value of the data segment   \*/  public int \_getData(int $location, int $offset)  {      return 0;  }    /\*\*   \* Store data   \*   \* Use offset 0 when it is just a single value.   \* Example: \_storeData($data,'data',1)   \* This will put the value of $data into the data segment "data" at position 1   \*   \* @param string $variable The variable to store   \* @param string $location The name of the location where the variable is stored   \* @param int    $offset   The offset of the location   \*   \* @return void   \*/  public void \_storeData(int $variable, int $location, int $offset)  {  }    } |