IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 3 Issue 1, January 2016.
www.ijiset.com
ISSN 2348 – 7968

Algorithms for Computing Earliest and Latest Dates of Offender Release

Sinyinda Muwanei,Douglas Kunda, Jacob Tangishi

Mulungushi University, Centre for ICT Education, P.O Box 80415, Kabwe, Zambia

Email: msinyinda@mu.ac.zm, dkunda@mu.ac.zm, jtangishi@mu.ac.zm

Abstract: - there are prisons in every nation of the world. Each of these require a standardized way of computing duration of stay of offenders after convictions of guilt have been arrived at by the courts of law. The way the computations will be carried out will depend on the type of sentence passed by the courts of law. This paper presents algorithms for computing initial earliest dates of release as well as latest release dates.

Keywords : - Jail Sentence, Algorithm, Software, offender management

1. Introduction

According to Dictionary of probation and offender management, offender management is the processes and arrangements for handling offenders from arrest through to the completion of sentences. An offender is an individual that commits acts that are illegal in society. These illegal acts are referred to as offences. According to the blackslaw dictionary an offender is “the name that is used for a person who is guilty of an offense according to law “. Examples of offences are murder, bribery, assault, felony and theft. After being convicted by the courts of law, it is necessary for the prisons to determine the date of release of offender based on the sentence passed by the courts of law. A sentence is simply the duration the offender will spend in prison. Prisons are segregative institutions designed to punish, typically for criminal offences. Prisons are usually strictly regulated and tightly surveyed environments, to which offenders are confined for varying lengths of time, and sometimes for life (Gaillard et al, 2012). The prison is a closed environment and there are laws,written ones which appear in formal documents and the unwritten ones which define how individuals should behave which person to cross which person to avoid(Ghazala B,2010). There are various types of sentences offenders serve in prison include cumulative and consecutive sentences.

- A **concurrent** sentence is served at the same time as another sentence imposed earlier or at the same proceeding.
- A **consecutive** (or **cumulative**) sentence occurs when a defendant has been convicted of several counts, each one constituting a distinct offense or crime, or when a defendant has been convicted of several crimes at the same time. The sentences for each crime are then "tacked" on to each other, so that each sentence begins immediately upon the expiration of the previous one (Types of Sentences 2015).

The other type of sentence is called the initial sentence. This is the first sentence that an offender is given when convicted of the first offence. for computing all these sentences it is necessary to consider the dates of conviction from the courts of law, the actual sentence in years, months or days as well as the type of offence committed by the offender. Lack of algorithm implementations especially in developing nations have led to many offenders being incarcerated for longer periods than their sentences. This problem is not only in developing countries but also developed ones. In the united states for example ,human error and outdated technology have miscalculated thousands of prison sentences and cost some states millions of dollars (States Turn to Technology to Calculate Prison Sentences,2015). The other challenge experienced by countries world over is over crowding. The prison population of the world is rising (Roger W. et al,2003) and According to the International Centre for Prison Studies (2004), overcrowding is common in the East and in the West, in the developing as well as the developed world. According to Walmsley(2008) over 9.8 million people are held in penal institutions throughout the world, mostly as pre-trial detainees or having

ISSN 2348 – 7968

been convicted and sentenced. The systematic procedure or algorithm and its implementation in a high level programming language for sentence computation may help with resolutions of the above mentioned problems. An algorithm is a tool for solving a well-specified computational problem. The statement of the problem specifies in general terms the desired input/output relationship. The algorithm describes a specific computational procedure for achieving that input/output relationship(Thomas H. et al,2009).

In this paper Java has been chosen as the implementation language. Java is an object oriented programming language that was developed by James Gosling at Sun Microsystems. It was released in 1995 as a part of Sun Microsystems' Java platform. (Deitel and Deitel, 2010). The language has developed much of its syntax from C and C++. Java applications are usually compiled to bytecode (class file) that can run on any Java Virtual Machine (JVM). (Gosling and McGilton, 1995).
Java is platform independent as it can run on many different operating systems. Java does this by making the Java compiler turn code into Java bytecode instead of machine code. This means that when the program is executed, the Java Virtual Machine interprets the bytecode and translates it into machine code. (Deitel and Deitel, 2010).
According to (Deitel and Deitel, 2010). Java was developed to achieve 5 main goals. These are:

- It should be simple, object-oriented, distributed and easy to learn.
- It should be robust and secure.
- It should be independent of a given computer architecture or platform.
- It should be possible to write an interpreter for the language. The language should also support parallelism and use dynamic typing.

There are many types of Java programs which run differently:

- Java Applet - is usually stored on a website and is downloaded and run on a client computer from within a web browser
- Application - can only be run on the computer. If online, it has to be downloaded before being run.
- JAR File - is used to package Java files together into a single file. (Almost exactly like a .zip file.)
- Servlet - runs on a web server and helps to display web pages.
- Swing application - is used to build an application that has a GUI (windows, buttons, menus, etc.)
- EJB - runs on a web server and is used to develop large, complex websites, and jsp.Java Server Pages (JSP) is a technology that helps software developers create dynamically generated web pages based on HTML, XML, or other document types. Released in 1999 by Sun Microsystems, JSP is similar to PHP and ASP, but it uses the Java programming language.
- To deploy and run Java Server Pages, a compatible web server with a servlet container, such as Apache Tomcat or Jetty, is required. (Hall, 2002).

2. Algorithms for computing of initial Sentences

There are several methods of specifying algorithms and two of the most common are flowcharts and pseudo code. Flowchart is a method of expressing an algorithm by a collection of connected geometric shapes containing descriptions of the algorithm's steps. This representation technique has proved to be inconvenient for all but very simple algorithms(Levitin anany,2012). Pseudo code is sometimes referred to as pseudo language. A form of representation used to provide an outline description of specification for a software module. Pseudo languages contain a mixture of natural language expressions embedded in syntactic structures taken from programming language. Pseudo languages are not intended to be executed by the computer ; they must be interpreted by people. (NCC Education,2008)

In this paper Pseudo code will be used since much detail needs to be outlined and the algorithm is not relatively simple.

2.1 Algorithm for Calculation of Earliest possible Date

The algorithm in pseudocode below is for calculation of earliest possible dates of offender release. The algorithm is a representation of a function with seven parameters. The table 1.1 shows the variables and their meanings.

ISSN 2348 – 7968

```
DATE FUNCTION EARLIEST_POSSIBLE_RELEASE(LPD,DOC OF TYPE STRING,day,month,int year, sentence OF TYPE INTEGER)
Use variables:
year_convert_months, year, month_remainder, day_remainder, months, months_total, days_total, month_oneThird, day_oneThird, day_temp,month_temp,year_temp,day_1,month_1,year_1,final_day, final_month, final_year,days,remissionNew,difference, max, diffInMillisec,
diffInDays,diff OF TYPE INTEGER ,
Doc,lpd, Earliest_Possible_Date OF TYPE STRING
cal1,cal2,cal3 OF TYPE CALENDAR
df OF TYPE DATE_FORMAT
 year_convert_months :=year*12
diff := 0.3 * sentence
difference:=sentence-diff;
months_total:=year_convert_months+month
month_oneThird:=months_total/3
month_remainder:=months_total%3
IF month_remainder=1
days=days+10;
ENDIF
IF month_remainder=2
days=days+20;
ENDIF
day_oneThird :=day/3+days
day_remainder :=day%3
IF day_remainder=2
day_oneThird=day_oneThird+1;
ENDIF
IF month_oneThird >=12
year_oneThird :=month_oneThird/12
ENDIF
month_oneThird :=month_oneThird%12;

 day_temp:=cal1.GET_DAY_OF_MONTH
 month_temp:=cal1.GET_MONTH_CALENDAR
 year_temp:=cal1.GET_YEAR_CALENDAR
IF day_oneThird > day_temp  THEN


    final_day:=30+day_temp-day_oneThird
    final_day:=final_day+1
    month_temp:=month_temp-1
 ELSE
```

ISSN 2348 – 7968

```
 final_day:=day_temp-day_oneThird
final_day:=final_day+1
ENDIF
   IF month_oneThird > month_temp  THEN
    final_month:=12+month_temp-month_oneThird
    year_temp:=year_temp-1
    ELSE
    final_month:=month_temp-month_oneThird
    year_temp:=year_temp
ENDIF
    final_year:=year_temp-year_oneThird
     cal2.SET_TIME(date_conviction)

dayTemp:=cal2.GET_DAY_OF_MONTH_CALEN
DAR
    monthTemp:=cal2.GET_MONTH_CALENDAR
    yearTemp:=cal2.GET_YEAR_CALENDAR
String                                  dT:=
yearTemp+"/"+monthTemp+"/"+dayTemp
EPD_:=df.CONVERT(dT)
max := cal2.GET_MAXIMUM_DAY_MONTH
diffInMillisec      :=          EPD_.GET_TIME      -
date_conviction.GET_TIME
diffInDays := diffInMillisec / (24 * 60 * 60 * 1000);
 IF diffInDays>=max THEN
  final_month:=final_month+1
  Earliest_Possible_Date:=final_year+"-
"+final_month+"-"+final_day
 ENDIF
 IF diffInDays<max

day_1:=cal2.GET_DAY_OF_MONTH_CALENDA
R
    month_1:=cal2.GET_MONTH_CALENDER
    year_1:=cal2.GET_YEAR_CALENDAR
    RemissionNew := 0.3* sentence
   sum_day:=sentence+day_1
   diffMax:=0
ENDIF
IF sum_day>max
sum_day:=sum_day-max-1
month_1:=month_1+1;
diffMax:=max-sum_day-1;
sum_day:=sum_day-RemissionNew+diffMax;
 ENDIF
RETURN Earliest_Possible_Date;

ENDFUNCTION
```

Variable Name	Type	Meaning
year_convert_months	integer	All month data held here after year conversion
year	integer	Year data as entered by user
month_remainder	integer	Holds remainder of division of month
day_remainder	integer	Holds remainder of division of day
months	integer	Month data as entered by user
months_total	integer	Holds months including the remainder from division of year
days_total	integer	Holds months including the remainder from division of month
month_oneThird	integer	Holds one third of total months as per algorithm requirement
day_oneThird	integer	Holds one third of total days as per algorithm requirement
day_temp	integer	Holds temporal value of day component during calculations
month_temp	integer	Holds temporal value of month component during calculations
year_temp	integer	Holds temporal value of year component during calculations
day_1	integer	Holds temporal value of day before calculation of final day
month_1	integer	Holds temporal value of month before calculation of final month
year_1	integer	Holds temporal value of year before calculation of final year
final_day	integer	Holds the final value of day component for the date
final_month	integer	Holds the final value of month component for the date
final_year	integer	Holds the final value of year component for the date
days	integer	Holds number of days as entered by user
remissionNew	integer	Holds remission value as per algorithm requirement
max	integer	Holds maximum number of days of month for conviction date
diffInMillisec	integer	Holds value in milliseconds of difference between conviction date and earliest possible date
diffInDays	integer	Holds value of difference in days between conviction date and earliest possible date

ISSN 2348 – 7968

diff	integer	Holds difference between max and total days
Doc	String	Holds date Initial date of conviction
lpd	String	Holds the latest possible valid date
Earliest_Possible_ Date	String	Holds earliest possible valid date
cal1	OBJECT	First calendar object for string to date conversion
cal2	OBJECT	Second calendar object for string to date conversion
cal3	OBJECT	Third calendar object for string to date conversion
df	OBJECT	Object for date formating

Table 2.1

2.2 Algorithm for Calculation of Latest possible Date

The algorithm below is for calculation of latest possible dates of offender release. The algorithm is a representation of a function with four parameters. The table 1.1 shows the variables and their meanings.

STRING FUNCTION LATEST_POSSIBLE_RELEASE(DOC **OF TYPE DATE**,day,month,int year **OF TYPE INTEGER**)
Use variables:
year,month,dayTemp, monthTemp, yearTemp,day **OF TYPE INTEGER** ,
lpd **OF TYPE STRING**
cal1 **OF TYPE CALENDAR**
df **OF TYPE DATE_FORMAT**

cal1.SET_TIME(DOC)
cal1.ADD_MONTHS(month)
cal1.ADD_DAYS(day)
cal1.ADD_YEARS(year)

dayTemp:=cal1.GET_DAY_OF_MONTH_CALENDAR
monthTemp:=cal1.GET_MONTH_CALENDAR
yearTemp:=cal1.GET_YEAR_CALENDAR
String dT:= yearTemp+"/"+monthTemp+"/"+dayTemp

ENDIF
RETURN dT

ENDFUNCTION

Variable Name	Type	Meaning
year	integer	Year data as entered by user
months	integer	Month data as entered by user
dayTemp	integer	Holds temporal value of day component during calculations
monthTemp	integer	Holds temporal value of month component during calculations
yearTemp	integer	Holds temporal value of year component during calculations
days	integer	Holds number of days as entered by user
Doc	String	Holds date Initial date of conviction
lpd	String	Holds the latest possible valid date
cal1	OBJECT	First calendar object for string to date conversion
df	OBJECT	Object for date formating

Table 2.2

ISSN 2348 – 7968

3. Implementation of algorithms in Java

3.1 implementation earliest possible date of conviction

```
public String Calculate_EPD_consecutive(String LPD,String DOC,int day,int month,int year,int sentence ){
                    int year_oneThird=0;
                    int year_convert_months =year*12;
                    int month_remainder=0;
                    int day_remainder=0;
                    int months=0;
                    int months_total=0;
                    int days_total=0;
                    int month_oneThird=0;
                    int day_oneThird=0;
                    int final_day=0 ;  //this variable holds the final day value
                    int final_month=0 ;  //this variable holds the final month value
                    int final_year=0 ;  //this variable holds the final year value
                    int days=0;
                    String doc=DOC;
                    String lpd=LPD;
                    String Earliest_Possible_Date="";
                    int max=0;
          long diffInMillisec=0;
                    long diffInDays=0;
```

```
//check if the day_oneThird is greater than DAY_OF_MONTH
int day_temp=now.get(Calendar.DAY_OF_MONTH);
int month_temp=now.get(Calendar.MONTH);
 int year_temp=now.get(Calendar.YEAR);

if(day_oneThird > day_temp )
     {
  final_day=30+day_temp-day_oneThird;
      final_day=final_day+1;// one day is added as is part of the algorithm
  month_temp=month_temp-1;  // we had removed 30 days earlier therefore substr
 }
     else
     {
      final_day=day_temp-day_oneThird;
      final_day=final_day+1;// one day is added as is part of the algorithm
 }
```

3.2 implementation latest possible date of conviction

The snippet of implementation below is for the latest possible date of release.

```
public String LatestPossibleDate(String dateConviction,String year,String mont
{
    String LPD="";
    //Declaration and Conversion of Year,Month and Day into integers
    int Day=Integer.parseInt(day);
    int Month=Integer.parseInt(month);
    int Year=Integer.parseInt(year);
    //Declaration and Conversion of Date of Conviction
    String DateConviction=dateConviction;
    /*
    String doc=DateConviction.replaceAll("-","/");// this replaces all - chara

    SimpleDateFormat sdf = new SimpleDateFormat("yyyy/MM/dd");
    SimpleDateFormat nt = new SimpleDateFormat("dd-MM-yyyy");
    */
    Calendar now = Calendar.getInstance();
```

.....

.....

The snippet of implementation below is for the earliest possible date of release. The important assumption is that the remission is one third of the sentence given to offender by relevant authorities such as the courts of law.

```
    Calendar ca = Calendar.getInstance();
         // set the year,month and day to something else
    ca.set(year_1,month_1,sum_day);
        // Addition of one day to the EPD is done here.
      int day1_=ca.get(Calendar.DAY_OF_MONTH);
      cal.add(Calendar.MONTH,+1);

  int month1_=ca.get(Calendar.MONTH);

  int year1_=ca.get(Calendar.YEAR);

      Earliest_Possible_Date=year1_+"-"+month1_+"-"+day1_;
                    }




      } catch (Exception ex) {

      }

   return Earliest_Possible_Date;
}
```

Figure 1

```
try {
   /* String dateOfconv =nt.format(sdf.parse(doc));
    String Dateofconviction=dateOfconv.replaceAll("-","/");*/
   //SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");
   SimpleDateFormat sdf = new SimpleDateFormat("yyyy/MM/dd");
   now.setTime(sdf.parse(DateConviction));
   now.add(Calendar.DAY_OF_MONTH,Day);
   now.add(Calendar.DAY_OF_MONTH,-1);
   now.add(Calendar.MONTH, Month);
   now.add(Calendar.YEAR, Year);
   int dayFinal=now.get(Calendar.DAY_OF_MONTH);
   int monthFinal=now.get(Calendar.MONTH);
   monthFinal++;
   int yearFinal=now.get(Calendar.YEAR);
   LPD=yearFinal+"-"+monthFinal+"-"+dayFinal;

     } catch (Exception ex) {

}
 return LPD;
}
```

Figure 2

ISSN 2348 – 7968

4.**0 Results of execution of algorithms**

Date of Conviction	Sentence			Earliest Possible Release Date	Latest Possible Release Date
	Year	Month	Day		
23/06/2004	1	0	0	23/02/2005	22/06/2005
05/03/2002	0	24	0	05/07/2003	04/03/2004
01/01/2000	0	0	10	11/01/2000	11/01/2000
11/06/1998	30	0	0	11/06/2018	10/06/2028
09/04/2001	30	36	0	09/04/2019	08/04/2031
04/04/2010	6	6	0	04/08/2014	03/10/2016
01/08/2012	0	0	24	25/08/2012	25/08/2012
07/03/2013	9	9	9	13/09/2019	15/12/2022
08/07/2014	9	36	18	20/07/2022	25/07/2026
11/11/2015	0	1	0	11/12/2015	11/12/2015

Table 4.0 results of execution of algorithm implementation

5.0 Conclusion

Accurate computations of durations of confinement of offenders in any nation is very critical. In this paper, algorithms and implementations of computations of earliest and latest possible release dates have been proposed and demonstrated with results which indicate acceptable degree of accuracy of the algorithm. There are several areas where this algorithm could be improved. The algorithm needs to take into account weekends and public holidays as releases of offenders are only done in week days. The next improvement would be considerations of hours since currently only years, months and days are taken into account.

References:

Deitel, P. and Deitel, H. (2010). *Java, late objects version*. Upper Saddle River, N.J.: Pearson/Prentice Hall

Gosling, J. and McGilton, H. (1995). *The Java language environment*. Mountain View, California: Sun Microsystems.

Hall, M. (2002). *More servlets and JavaServer pages*. Upper Saddle River, NJ: Prentice Hall.

International Centre for Prison Studies (2004),*Guidance Notes on Prison* Reform Kings College, London.

Levitin Anany (2012). *Introduction to the design and analysis of algorithms*. Pearson Education

NCC Education (2008) . *Programming Methods* . NCC Education Limited

States Turn to Technology to Calculate Prison Sentences 2015 available from : < http://www.governing.com/columns/tech-talk/gov-states-technology-compute-prison-sentences.html>

Thomas H. Cormen,Charles E. Leiserson,Ronald L. Rivest & Clifford Stein (2009) *Introduction to algorithms*, MIT Press

Types of Sentences 2015 available from : <http://criminal.findlaw.com/criminal-procedure/types-of-sentences.html#sthash.TPFBHsSb.dpuf>

Walmsley Roy 2000, *World Prison Population List*. In Research Findings No. 116, Home Office Research, Development and Statistics Directorate, Centre for Prison Studies, London.

J.C. Gaillard, Fanny Navizet, Prisons, prisoners and disaster, International Journal of Disaster Risk Reduction, Volume 1, October 2012, Pages 33-43

Ghazala Bhatti, Learning behind bars: Education in prisons, Teaching and Teacher Education, Volume 26, Issue 1, January 2010, Pages 31-36

Roger Watson, Anne Stimpson, Tony Hostick, Prison health care: a review of the literature, International Journal of Nursing Studies, Volume 41, Issue 2, February 2004, Pages 119-128

ISSN 2348 – 7968

Biography

Mr Sinyinda Muwanei is currently Lecturer in the Centre for ICT Education at Mulungushi University in Kabwe, Zambia.He holds a Master degree from St Petersburg, RUSSIA. He has vast experience in in Java, C++, PHP/ MySQL web platforms, Microsoft products including SQL Server databases, Oracle and Postgresql databases and CISCO Networking

Dr Douglas Kunda is currently the Director for the Centre for ICT Education at Mulungushi University in Kabwe, Zambia. He holds a Doctorate degree in Computer Science from the University of York, UK. He is Fellow of the Computer Society of Zambia and Member of Association for Computing Machinery. He worked as the Project Manager for the Integrated Financial Management Information System (IFMIS) project for the Ministry of Finance. He has presented papers at International Conferences and published in journals. He is certified SAP ERP Solution Manager Consultant with experience in Java, C++, PHP/ MySQL web platforms, Microsoft products including SQL Server databases, Oracle and Postgresql databases, Linux, CISCO networking, PRINCE2 and moodle.

Mr Jacob Tangishi is currently pursuing his bachelors degree in Computer Science at Mulungushi University in Kabwe, Zambia. He has good experience in in Java, Microsoft products including SQL Server databases and Oracle Databases