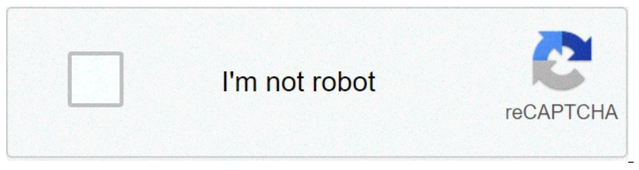

Continue

Army julian date and time

TM 11-5820-890-10-1 A-2 JULIAN DATE CALENDAR (REGULAR ARNO) JULIAN DATE CALENDAR (LEAP YEAR) The above module is initialized at the universal date and time (at all intent and purposes, this is the same as Greenwich Mean Time, but not exactly the same - see differences,) as determined by the clock in the computer at the time this page entered (Note: this is always the second nearest). A conversion from your local time zone is applied. The clock above the module shows the current universal time according to the Time and Date server clock, and will keep 'ticking'. The time indicated in the non-tick module - this and date are free for you to change as input to Julian's date calculator. The fields that have a blue/gray background are read-only. The field of the week is determined by the date of Julian and the time zone field is fixed according to the date and local time when the session was started. This includes any correction for “light saving of the day” (or “summer time”). EC and ECB designate “Common Era” and “First Common Era”, often known as “AD” and “BC”. See notes below for year-of-use option zero. The UNIX time stamp is used in the programming language ‘C’ and to show the expiration dates of cookies in Mozilla Firefox, among other things. It is defined as the number of seconds since 1970-Jan-01 00:00:00 UTC. It is often set to be a whole 32-bit; if so, fractions are truncated. If set to be ainteger, a system will not manage the times before January 1, 1970 correctly, and the last time it will be expressed correctly will be 2106-Feb-07 06:28:15 UTC. More concern could be the software using an entire signed, so a bug type ‘Y2K’ will cause failure after 2038-Jan-19 03:14:07 UTC. However, this calculator shows all positive times correctly, plus fractions (to 10m closer). The times before 1970 are indicated as negatives. File Time (MFT) is a 64-bit integer used by Microsoft to show the number of 100ns ‘ticks’ from 1601-Jan-01 00:00:00 UTC. It is often divided into two whole 32-bit (high and low), each of which is expressed in decimal form, as in this calculator. It is mainly used in Internet Explorer cookies to indicate expiration and creation times, and also for time stamping files in NTFS file systems, such as they are used in Windows 2000 and later. The 64-bit MFT integer will break in the year 60056, so it will be valid for the foreseeable future! Since the “tick” is only 100ns, which is 1/100000th of 10ms (the maximum resolution of this calculator), the 5 less significant data in the bottom entire are meaningless here, so they are set to zero. Moreover, since MFT cookies before 1601 CE would be a bit stale, negative results are not given. While the UNIX start date in 1970 seems reasonable, since the programming language ‘C’ was developed in the early 1970s, it is a mystery for which Microsoft chose 1601 for the MFT - thoughIt's not the only mysterious thing about Microsoft! [Wikipedia gives a somewhat subtle reason, in my opinion, that this date was chosen because it was the beginning of the 400-year Gregorian bisexual cycle within which the digital files before existed.] NB: If the expiration time of the cookie decoding using this calculator, the time indicated may differ from the time reported by the browser of a few hours. This will be due to the difference between the local time and UTC when the cookie is set to expire. [It is a MFT feature that the value is stored as UTC, but is translated locally when viewed. Maintaining everything in UTC means that it is impossible for a cookie or file to read before it was written - something that could happen to the files transmitted on the Internet between different time zones if local times were used.] The day of Mayan Long Calendar is shown here (read only) because the date 2012-Dec-21 was significant. On this date, the second box from the left (b’ak’tun) was moved to 13, which was the same Maya date as the beginning of the Fourth Creation. Because of this, and almost certainly because it was also the northern winter solstice, some condemnation prophets have calculated that the world would end that day. All this is explained long by Wikipedia. Well, the world didn't end then: Maya philosophy was instead that a re-creation would take place. We hope that the Fifth Creation will begin better than the Fourth End.... Note that scholars and astronomers may not agreethe oo of the Julian calendar for dates in the past (Mayan scholars treat all dates of the conventional calendar in the past as Gregorian, using the 'proleptic Gregorian calendar'.) therefore the dates before the Gregorian change (in 1582) which are given in the wikipedia article differ from those calculated here. to agree with the wikipedia article, the b’ak’tun value is never null for the dates of the past: the beginning of the fourth creation (jd 584283) is noted as 13.0.0.0.0.0.0, and the next b’ak’tun (1.0.0.0.0 = jd 728283) began after 13.19.17.19 (jd 7282.) previous roll-over had b’ak’tun values between 1 and 13. to compete with the representation of Mayan scholars of future dates, the figure b’ak’tun will now rise to 19 and then folds to zero, when the figure more significant later (piktun) becomes a 1. even if the piktun will go to 0 after 19.19.19.19.19.19.19.19.19.19.19.19.19.19.19.19.19.19.19.19.19.19.19.19.19 will be jd 58184282 or 154591-Feb-27. This utility is based on a calculator that was originally published by the department of astronomical applications of the naval observatory of the United States, but that version did not properly manage local times east of London. However, the content of that site (and url within it) continues to change, quickly making out-of-date any links I put here. then the link above is only at their home page: If you want to see how they do it now, look upsite (but please come back here.!) This calculator uses the original format to convert from calendar date and time to Julian Date, and vice versa. The conversion direction is selected by the calculation type. This page and its associated JavaScript file were extended by Steve Glennie-Smith to increase accuracy to the nearest 10m. Other extras include a ‘ticking’ UTC watch; the calculation of and from UNIX time stamp and Microsoft file format; and the display of the local time zone and the Mayan long calendar date. Bug fixes are listed at the end. Julian's dates (abbreviated JD) are simply a contiguous count of days and fractions from the noon Universal Time on January 1, 4713 BC (on Julian calendar). Almost 2.5 million days have passed since that date. Julian's dates are widely used as time variables in astronomical software. Typically, a 64-bit variable (double accuracy) can represent an epoch expressed as a Julian date at about 1 millisecond accuracy. Note that the time scale which is the basis for Julian's dates is Universal Time, and that 00:00h UTC always occurs on a date of Julian fraction of 0.5. The distinction between the date of Giuliano and the day of Giuliano is that the first is the entire number, including the fractional part, while the latter is only the whole part, that is. the number of days from the ‘zero day’. Why was he chosen on January 1, 4713 BC as a starting point? Some theories here. Calendar dates – year, month and day – are alsof a problem. Several calendar systems have been in use in different times and places around the world. This calculator only treats two: the Gregorian calendar, now used universally for civil purposes, and the Julian calendar, its predecessor in most of the Western world. As used here, the two calendars have identical names of the month and number of days in each month; They differ only in the rule for leap years. The Julian calendar has a leap year every quarter year, while the Gregorian calendar has a bisexual year every quarter year except for a hundred years that are not exactly divisible by 400. Although JD zero was on 1 January 4713 BC, this calculator will allow Julian Date negative, and years with up to six digits. The following assumptions are made: The weeks have always had seven days, as now, since time began. So, the day of the week can be obtained from the rest, after having divided the date of Julian (plus 0.5 day compensated, but then expressed as integer) from 7 (0 corresponds to Monday); The Julian calendar was no year. Therefore, 0001-Dec-31 ECB (or BC) was immediately followed by 0001-Jan-01 EC (or AD). However, some astronomers recognize a year 0. If this change is required, check the 'Use Year Zero' check box; The Julian calendar is always assumed to have had bisexual years. Since it was not year 0, years 1, 5, etc. ECB are considered leap years from this computer (if the year of zero use is unchecked). However, some research (see below)that before the introduction of the Julian calendar in 45 BC, there was no consistency in the number of days in each year; this calculator does not take second leaps into consideration. from 00:00h utc always occurs at a date of julian fraction of 0.5, it is assumed that the fractional part (which represents hours, minutes, seconds and parts of it) of julian dates where a second jump is inserted must be 'stretched' on such dates. Therefore the calculations made for such dates will be imprecise up to a second. since the second bisexuals are inserted only (or theoretically, they could be eliminated) at midnight utc, this will always occur at the center of a Julian day. (note: the purpose of a second jump is to correct for small variations in the time it takes for the earth to rotate on its axis. This has no weight on the number of days that it takes for the earth to orbit the sun, nor any correction for changes in this as applied by the Gregorian calendar;) the passage from the Julian calendar to the Gregorian calendar took place in October 1582, according to a papal bubble issued by Gregorian pope xiii. specifically, for dates of 4 October 1582, the Julian calendar is used; for dates of 15 October 1582 the Gregorian calendar is used. Thus, there is a gap of ten days on calendar dates, but no discontinuity on dates or days of julian of the week: on 4 October 1582 (julian) was a Thursday, which began in Jd 2299159.5; and 15 October 1582 (gregorian)a Friday, which began at JD 2299160.5. It was necessary to delete ten calendar dates due to the error accumulated by the Julian calendar: in many centuries of use, there had been too many leap years. The passage to the Gregorian calendar occurred only as described above in Roman Catholic countries, however. The adoption of the Gregorian calendar in the rest of the world progressed slowly. For example, Britain and its colonies did not change until September 1752 (although there is some doubt about when the change took place in Scotland). [The UNIX CAL command reflects the change of 1752, when it became necessary to eliminate eleven days, from the 1700s it was also taken as a bisexual year.] This has produced some interesting anomalies. For example, Spanish author Miguel Cervantes and English playwright William Shakespeare both died on the same date (23 April 1616), but not on the same day! This is because Spain had adopted the Gregorian calendar, but England had not yet done so. The corrections applied by the Gregorian calendar give a close average approximation (365.2425 days) when it takes us on earth to orbit the sun. However, the real time taken (averagely over the last centuries) is 365,242190 days, so a small error still matures, although now it is only a gain of 1 day in about 3220 years. Several extensions were proposed to the rules of 100 and 400 years in the Gregorian calendar, but no one was officiallya rule of 4000 years: astronomer John herschel suggested that if the year is divisible within 4000 should not be a bisexual year. this gives a better approximation having 969 leaps in every 4000 (not 970,) giving an average year of 365,24225 days. even if not the most accurate of the proposed corrections, falls neatly within the sequence 4 - 100 - 400. the unix cal command includes this correction. a rule of 3200 years, similar to the rule of 4000 years above: this is much closer to facing the discrepancy of 3220 years, but it seems to have little support outside the army. the correct average year would be 365.2421875 days. a rule of 128 years: this is exhausted with the rules of 100 and 400 years of all. the proposal is to fall on 29 February for divisible years of 128, which begins in 2048. is as accurate as the 3200 rule and correction is applied more evenly. could be implemented at any time before 2048 without need for re-sync because the required number of “non-year” has already been omitted. [1920 would not have been a leap year but 1900 would have been; in the same way 1792 ~ 1800 and 1664 ~ 1700. 1536 and before they were in the Julian era, and so would not have been interested.] although great for computer geeks and those who love 2 powers, it would not be so easy for ordinary mortals to manage. by I will be 100 in 2048, if I am still around then I could expect my real telegram one day before if this system is adopted inTime! A 900-year rule: This was proposed by the Greek Orthodox Church. It moves away with the rule of 400 years and instead years that leave a rest of 200 or 600 when divided by 900 become bisexual years. The correct average length of the year would be 365.242222 days. Both this and the Gregorian system give 2000 and 2400 as bisexual years. The differences begin in the 2800, which would not be a bisexual year in the Greek system, but 2900 would be. In my view, it would be absurd to impose changes that would have taken place so far on future generations. Who knows – an idiot could leave an atomic bomb or the land could be hit by a great meteorite, both could affect the length of the year, and thus blow all this theory from the water. However, Julian's dates will continue to flatter independently. Since the goal of this calculator is to give an accurate correlation between julie dates and calendar given way in the future, I agreed with Herschel and UNIX, and modified the original JavaScript to include rule 4000 years. More information about when various countries can be found on the calendar pages of this European history site. For another list of when some countries went to the Gregorian calendar, see Section 2.2.4 of the Claus Tøndering FAQ calendar. Further information on the calendars and their stories is found in the chapter “calendari” of L.E. Doggett of the explanatory supplement to the astronomical almanac (ed. P.K. Seidelmann, 1992,Scientific books). Note: These are commercial libraries sites. Note: There are a number of other data giulia calculators on the web: a quick search on Google presented this, which works very slowly and is confused by the dates of the ECB. Presumption (as a number of others) that all calendar dates are Gregorian and, according to it, JD zero was BCE4712 (month 0) (day -2)! I had a connection to a worse one, which was wrong before 1800 and after 2100 CE. It has since gone, but there are probably others. Attention! Bug fixes: Conversion to Universal Time for time zones east of London (and for the United Kingdom, Ireland and Portugal when BST is in force) has not always been shown correctly. In particular, if the application is inserted after midnight of the club, but before midnight of Universal Time, the local date has been displayed (i.e. one day after it should be), along with a negative time. The time in time zones of half an hour, such as India and Central Australia, was not displayed correctly. The U.S. Naval Observatory later avoided this problem by setting the server date (more than local) (the time is always set at midnight), but I think the maintenance of initialization at the date and local time is more useful. Also, since the calculation code now appears to be run server side, it is no longer visible to the user. I left the associated JavaScript file for this publicly available computer - it should be in the browser cache as . You're welcome.use it in other applications, but a recognition would be appreciated. If the local clock is fixed at a date of the 20th century, sometimes a date of the 21st century was initialized (e.g. 2098 instead of 1998). This is because some browsers interpret the 2000s and then differently in their JavaScript interpreters - see for more details. Where the integer part of JD was small enough to allow more than 5 decimal points, any fraction between 0.4995 and 0.49999999999999999999 caused the wrong weekday display or a JavaScript error. Steve Glennie-Smith 28 Dec 2012 what is today's julian date for military. why does the military use julian date