```
public class UniqueChars {
    public static void main(String[] args) {
        String input = args[0];
        int length = input.length();
        String str = "";
        for (int i = 0; i < length; i++) {
            char currentChar = input.charAt(i);
            if (str.indexOf(currentChar) == -1 || currentChar == ' ') {
                str = str + currentChar;
            }
        }
        System.out.println(str);
    }
}
```

```
public class LowerCase {
    public static void main(String[] args) {
        String input=args[0];
        int length=input.length();
        String str="";
        for (int i = 0; i <length; i++) {
            if(input.charAt(i)>='A' && input.charAt(i)<='Z'){
                char chr= (char) (input.charAt(i)+32);
                str=str+chr;
            } else{
                str=str+input.charAt(i);
            }
        }
        System.out.println(str);
    }
}
```

```
public class  LoanCalc {
	static double  epsilon = 0.001;
	static int iterationCounter=0;
	public static void main(String[] args) {
		double loan=Double.parseDouble(args[0]);
		double rate = Double.parseDouble(args[1]);
		int n = Integer.parseInt(args[2]);
		System.out.println("Loan sum = " + loan + ", interest rate = " + rate +
"%, periods = " + n);
		System.out.print("Periodical payment, using brute force: ");
		System.out.printf("%.2f", bruteForceSolver(loan, rate, n, epsilon));
		System.out.println();
		System.out.println("number of iterations: " + iterationCounter);
		iterationCounter=0;
		System.out.print("Periodical payment, using bi-section search: ");
		System.out.printf("%.2f", bisectionSolver(loan, rate, n, epsilon));
		System.out.println();
		System.out.println("number of iterations: " + iterationCounter);
	}
	public static double bruteForceSolver(double loan, double rate, int n, double
epsilon) {
		double g=loan/n;
		while (endBalance(loan,rate,n,g)>0){
			g= g+epsilon;
			iterationCounter++;
		}
		return g;
	}
	public static double bisectionSolver(double loan, double rate, int n, double
epsilon) {
		double high = loan;
		double low = loan / n;
		double payment = (high + low) / 2;
		while (high - low > epsilon) {
			if (endBalance(loan, rate, n, payment) * endBalance(loan, rate,
n, low) > 0) {
				low = payment;
			} else {
				high = payment;
			}
			payment = (high + low) / 2;
			iterationCounter++;
		}
		return payment;
```

```
	}
	private static double endBalance ( double loan, double rate, int n, double
payment){
		double endBalance = loan;
		for (int i = 0; i < n; i++) {
			endBalance = (endBalance - payment) * (1 + rate / 100);
		}
		return endBalance;
	}
}
```

```
public class Calendar0 {
	public static void main(String args[]) {
		int year = Integer.parseInt(args[0]);
		isLeapYearTest(year);
		nDaysInMonthTest(year);
	}
	private static void isLeapYearTest(int year) {
		String commonOrLeap = "common";
		if (isLeapYear(year)) {
			commonOrLeap = "leap";
		}
		System.out.println(year + " is a " + commonOrLeap + " year");
	}
	private static void nDaysInMonthTest(int year) {
		for (int i = 1; i < 13; i++) {
			System.out.println("Month " + i + " has " + nDaysInMonth(i, year)
+ " days");
		}
	}
	public static boolean isLeapYear(int year) {
		if (year % 400 == 0 || (year % 4 == 0 && year % 100 != 0)) {
			return true;
		}
		return false;
	}
	public static int nDaysInMonth(int month, int year) {
		if (month == 1 || month == 3 || month == 5 || month == 7 || month == 8 ||
month == 10 || month == 12) {
			return 31;
		} else if (month == 4 || month == 6 || month == 9 || month == 11) {
			return 30;
		} else if (month == 2 && isLeapYear(year)) {
			return 29;
		} else if (month == 2) {
			return 28;
		}
		return 0;
	}
}
```

```
public class Calendar1 {
	// Starting the calendar on 1/1/1900
	static int dayOfMonth = 1;
	static int month = 1;
	static int year = 1900;
	static int dayOfWeek = 2;	// 1.1.1900 was a Monday (2nd day as
sunday,monday)
	static int nDaysInMonth = 31; // Number of days in January
	public static void main(String args[]) {
		int debugDaysCounter = 0;
		String isSunday="";
		int firstSundayCounter=0;
		while (true) {
			isSunday="";
			if(dayOfWeek==1){
				isSunday = " Sunday";
				if(dayOfMonth==1){
					firstSundayCounter++;
				}
			}
			System.out.println(dayOfMonth+ "/"+month+"/"+year+isSunday);
			advance();
			debugDaysCounter++;
			if (debugDaysCounter==36524) {
				System.out.println();
				System.out.println("During the 20th century, "+
firstSundayCounter+" Sundays fell on the first day of the month");
				break;
			}
		}
	}
	private static void advance() {
		nDaysInMonth = nDaysInMonth(month, year);
		dayOfWeek++;
		dayOfWeek = dayOfWeek % 7;
		dayOfMonth++;
		if (dayOfMonth > nDaysInMonth) {
			month++;
			dayOfMonth = 1;
		}
		if (month > 12) {
			month = 1;
			year++;
		}
```

```
	}
	public static boolean isLeapYear(int year) {
		if (year % 400 == 0 || (year % 4 == 0 && year % 100 != 0)) {
			return true;
		}
		return false;
	}
	public static int nDaysInMonth(int month, int year) {
		if (month == 1 || month == 3 || month == 5 || month == 7 || month == 8 ||
month == 10 || month == 12) {
			return 31;
		} else if (month == 4 || month == 6 || month == 9 || month == 11) {
			return 30;
		} else if (month == 2 && isLeapYear(year)) {
			return 29;
		} else if (month == 2) {
			return 28;
		}
		return 0;
	}
}
```

```
public class Calendar {
	static int dayOfMonth = 1;
	static int month = 1;
	static int dayOfWeek = 2;	// 1.1.1900 was a Monday (2nd day as
sunday,monday)
	static int nDaysInMonth = 31; // Number of days in January
	static int year=1900;
	public static void main(String args[]) {
		int newYear = Integer.parseInt(args[0]);
		int debugDaysCounter = 0;
		String isSunday="";
		int firstSundayCounter=0;
		while (true) {
			isSunday="";
			if(dayOfWeek==1){
				isSunday = " Sunday";
				if(dayOfMonth==1){
				}
			}
			if(year==newYear) {
				System.out.println(dayOfMonth + "/" + month + "/" + year
+ isSunday);
			}
			advance();
			debugDaysCounter++;
			if (year>newYear) {
				break;
			}
		}
	}
	private static void advance() {
		nDaysInMonth = nDaysInMonth(month, year);
		dayOfWeek++;
		dayOfWeek = dayOfWeek % 7;
		dayOfMonth++;
		if (dayOfMonth > nDaysInMonth) {
			month++;
			dayOfMonth = 1;
		}
		if (month > 12) {
			month = 1;
			year++;
		}
	}
```

```
	public static boolean isLeapYear(int year) {
		if (year % 400 == 0 || (year % 4 == 0 && year % 100 != 0)) {
			return true;
		}
		return false;
	}
	public static int nDaysInMonth(int month, int year) {
		if (month == 1 || month == 3 || month == 5 || month == 7 || month == 8 ||
month == 10 || month == 12) {
			return 31;
		} else if (month == 4 || month == 6 || month == 9 || month == 11) {
			return 30;
		} else if (month == 2 && isLeapYear(year)) {
			return 29;
		} else if (month == 2) {
			return 28;
		}
		return 0;
	}
}
```