Assignment 2

Introduction

This problem is inspired by an old paging standard called POCSAG (Post Office Code Standardization Advisory Group). At the time when this paging system was prevalent, electronics was not that advanced and the main consideration was to design pagers that could work for a reasonable amount of time before battery replacement. POCSAG achieves this by using time slots. Each pager is assigned a unique address, which also determines a unique starting time slot for the pager. Pagers sleep till a timer wakes it up just before its time slot. The pager will check if the time slot (assigned to it) contains a message intended for it. If so, it starts to read the message till the message end. After reading the message, the pager alerts the user, displays the message and goes back to sleep.

At the transmission end, the system receives a number of messages intended for different pagers. The system packs these messages into frames in such a way that the message for a specific pager will always start at the time slot corresponding to the pager. If the system does this packing blindly, then it could result in a lot of inefficiency, since after a message is transmitted, it must wait for the starting slot of the next message before it starts sending the next message. The gap between end of one message and the start of the next is wasted air time and air time is the most precious commodity for the paging company, since it licenses spectrum at a high price.

Definitions

A ***time slot*** is a fixed period of time within which one character can be transmitted along with a single bit that indicates whether the character is the first character of a message or not
A ***frame*** is a sequence of 128 time slots numbered 0 to 127.
A ***transmission*** T is a sequence of frames transmitted as one burst. Size(T) is the number of frames contained in the transmission T. The position of a frame in the transmission is called its frame number, which ranges from 0 to Size(T)-1.
A ***message*** is a pair (n, w) where n is a time slot and w is a string of characters. Length(w) is the number of characters in w. w will always satisfy: $6 \leq \text{Length}(w) \leq 48$.
Messages is the set of messages that have arrived and are awaiting transmission.
An ***allocation*** A assigns frame numbers to messages in Messages such that:

- If $m=(n,w) \in$ Messages, and $A(m)=\text{Size}(T)-1$, then $n+\text{Length}(w) \leq 127$ (i.e. no message overflows the transmission T)
- If $m1=(n1,w1)$, $m2=(n2,w2) \in$ Messages,
  then $n1+\text{Length}(w1) \leq n2$, if $A(m2) = A(m1)$ and $n1 < n2$, and
  $(n1+\text{Length}(w1)) \bmod 128 \leq n2$, if $A(m2) = A(m1) + 1$,

(i.e. the messages don't overlap. Each slot is occupied by at most one message)

The ***efficiency*** of an allocation A is OccupiedSlots/TotalSlots, where OccupiedSlots is the sum of Length(w) over all messages $(n,w) \in$ Messages, and TotalSlots = 128*Size(T).

The packing problem

Given a set Messages, find an allocation A that maximises the efficiency, i.e. uses the least number of frames in the transmission.

Input file

The input file (stdin) will have in the first line two numbers N and M separated by a single space. N represents the number of messages and M the expected number of frames within which these messages are to be packed. Each subsequent line has two numbers S and L separated by a single space, where S is the starting slot of a message and L is the length of the message. Line k+1 contains message number k and there are exactly N message lines.

Output file

The output file (stdout) will contain a single line with the word PASS or FAIL. The program should print PASS if it can pack the messages given in the input file within the expected number of frames. If it cannot it will print FAIL.

Examples

Input file:
1 1
125 10

Output file:
FAIL

Explanation: Message 1 overflows frame 1 and needs a second frame to complete

Input file:
2 1
80 45
0 44

Output file:
PASS

Explanation: Message 2 is sent first and then message 1 in frame 1. Efficiency is 89/128.

Input file:
3 2
6 45
12 47
90 44

Output file:
PASS

Explanation: Message 1 and 3 are sent in the first frame. Message 2 is sent in the second frame. Efficiency is 136/256.

Input file:
3 2
6 45
12 47
30 44

Output file:
FAIL

Explanation: No two messages can be placed in the same frame, so we need three frames.

Submission

Each submission will carry three components 5 test cases, the program and a 4 page report. Of the 5 test cases, 1 test case will be a FAIL test case and the other 4 should be PASS test cases. The 4 pass test cases should be in files named YOUR_ROLL_NO_PASS_1.in, YOUR_ROLL_NO_PASS_4.in and the FAIL test case should be in a file named YOUR_ROLL_NO_FAIL.in. The program should be in a file named YOUR_ROLL_NO.ext, The 4 page document should be in a pdf file and named YOUR_ROLL_NO.pdf.