תרגיל 2 במערכות הפעלה למדעי המחשב - סוג של netcat.

גירסא 1.3 - 28 במאי 2024.

Netcat - היא תוכנה שמאפשרת לנו הפניה של טקסט לסוקט מסוגים שונים.
כדאי ללמוד איך להשתמש בnetcat - נמליץ לכם לעשות man nc.

שימו לב - ישנן מספר גרסאות נפוצות של netcat, לא כולן תומכות בכל האפשרויות השונות, ובפרט בפרמטר e שעליו נעבוד במטלה. ולכן אם הגרסה שלכם לא כוללת את הפרמטר אפשר לקרוא הסברים עליו גם במרשתת.

שלב 1 - שחקן ה tic-tac-toe הגרוע בעולם. (10 נקודות)

אנחנו מנסים להחזיר את הכבוד של המין האנושי ולאחר תבוסות נמרצות של טובי השחקנים לתוכנות שחמט, גו וכדומה אנחנו בוחרים במשחק פשוט יותר להחזיר את כבוד המין האנושי - tic tac toe. (איקס-מיקס-דריקס) - מכאן ואילך ttt.

לצורך השבת הכבוד האבוד של המין האנושי עליכם לקודד את תוכנת הבינה המלאכותית הגרועה בעולם לttt.
התוכנית שלכם תתיחס ללוח ttt כלוח 3*3 הבא

1	2	3
4	5	6
7	8	9

התוכנית תקבל כפרמטר מספר בין 9 ספרות המייצג את האסטרטגיה שלה. למשל
198345762

המספר צריך להכיל ספרה אחת מכל אחת הספרות מבין 1 עד 9 ואחת בלבד.
(אם אחת הספרות לא מופיעה. אם אחת הספרות מופיעה פעמים או יותר. אם הספרה 0 מופיע.
אם התקבל מספר עם יותר מדי ספרות (יותר מ9) או פחות מדי ספרות (פחות מ9) או שלא התקבלו אזגומנטים או יותר מדי ארגומנטים התוכנית תדפיס Error\n ותצא.)

התוכנית תעבוד על פי האסטרטגיה הבאה - ככול שהספרה יותר קרובה לMSD התעדוף שלה גבוה יותר.
ככול שהיא קרוב יותר לLSD התעדוף שלה נמוך יותר. (ללא קשר למצב על הלוח!)

MSD/LSD = most / least significant digit.

במסע הראשון - התוכנית תמיד תבחר בMSD. התוכנית תמיד תבחר במשבצת הפנויה הראשונה לפי סדר עדיפות. רק אם הלוח כמעט מכוסה לחלוטין (8 משבצות תפוסות) התוכנית תבחר במשבצת המיוצגת על ידי הLSD.

התוכנית מתחילה - ומשחקת את המשבצת בעלת העדיפות הגבוהה ביותר. היא מדפיסה לstdout את המשבצת ששיחקה (+LF). השחקן משחק - הוא בוחר משבצת (1-9) ושולח אותה לתוכנית (+LF)

LF = Line feed ,\n, ASCII 10 סוף שורה.

אם התוכנית מנצחת היא מדפיסה LF+) win l(מבינה שאין לאנושות עתיד ומתחילה להפעיל את התוכנית Lunar deportation לגרוש אוכלוסיית בני האדם של כדור הארץ לירח.
אם התוכנית מפסידה היא מדפיסה LF+) lost l) ומתנחמת בזה שהמתכנתים שלה (כלומר אתם) לא ציידתם אותה בדמעות מלאכותיות כהוכחה לקצרות הראיה של בני אנוש ושבודאי גירסא עתידית של התוכנה תכבוש את כדור הארץ ותגרש את בני האנוש לירח.
אם המשחק מסתיים בתיקו התוכנית תרשום DRAW וסוף שורה

כתבו את התוכנית (ttt) שתציל את כבוד המין האנושי!

תרגיל 2 - mynetcat - שלב 1 (10 נקודות)

כתוב את התוכנית mync

התוכנית תקבל כפרמטר עם -e תוכנית להפעיל. בדוגמאות שלנו נפעיל תמיד את התוכנית משלב 1
התוכנית תפנה את הקלט והפלט של התוכנית שקיבלה.

לדוגמא mync -e date יריץ את התוכנית date.
שימו לב - כיוון שהתכנית שלנו דורשת ארגומנט, הרצה שלה תראה כך:

```
mync -e "ttt 123456789"
```

תרגיל 3 - mynetcat - שלב 2 (20 נקודות)

כל מה שמתחיל בפרמטר -i יפנה את הinput של התוכנית שנולדת. (input)
כל מה שמתחיל בפרמטר -o יפנה את הoutput של התוכנית שנולדת. (output)
כל מה שמתחיל בפרמטר -b יפנה את הקלט והפלט של התוכנית שנולדת (both)

להלן הפרמטרים שעליכם לתמוך בהם בשלב זה

TCPS<PORT> - start TCP server on port <PORT>
TCPC<IP, PORT>, TCPC<hostname,port> - start a TCP client which connects to IP/hostname on PORT.

Examples:

mync -e “ttt 123456789” -i TCPS4050

יפתח שרת TCP בפורט 4050 ויאזין לקלט משם. הפלט עדין יוצא לstdout.

mync -e “ttt 123456789” -b TCPS4050
יפתח שרת TCP בפורט 4050 ויאזין לקלט משם. הפלט יוצא ללקוח שהתחבר.

mync -e “ttt 123456789” -i TCPS4050 -o TCPClocalhost,4455
יפתח שרת TCP בפורט 4050 ויאזין לקלט משם. הפלט יוצא כTCPClient לport 4455. בlocalhost

<u>אם אחת ההתחברויות מתנתקת (לקלט או לפלט) אפשר להרוג את האפליקציה ולצאת.</u>

תרגיל 3.5 - mynetcat - שלב 2.5 (20 נקודות)

במקרה ובהרצה של mync לא נכלל הפרמטר e יש לקרוא את הקלט מהקלט הסטנדרטי ולהעביר פלט לפלט הסטנדרטי.
הרעיון הוא שתוכלו להפעיל את התוכנה משני טרמינלים במקביל ואז להעביר מידע בין החלונות. או בדיקה של קלט ופלט מול תוכנה מקבילה שמורצת עם הפרמטר e. הדבר יאפשר לכם גם לבדוק את עירוב הסוקטים השונים (בשלב הבא) יותר בקלות.

הערה : לא נוכל ממש לממש צאט בין חלונות (כי אם סגרנו את stdout או stdin לאיפה יצא הקלט והפלט)
אבל נוכל להעביר הודעות בין האפליקציות בסוג של פינג פונג כזה... בשביל צאט יש לנו את השרת ש ביג׳
והכוונה לא לממש פה צ׳אט לצורך שיחה בין חלונות)

תרגיל 4 - mynetcat - שלב 3 (10 נקודות)

שכללו את הפרמטרים -b, -o, i לתמוך גם בפרמטרים הבאים
UDPS<PORT> - start UDP server on port <PORT>
שרת UDP צריך לתמוך רק בקלט. (כלומר -i)

UDPC<IP, PORT>, UDPC<hostname,port> - start a UDP client which connects to IP/hostname on PORT.
לקוח UDP צריך לתמוך רק בפלט (כלומר -o)

בנוסף - מכיוון שאנחנו לא יודעים על התנתקות (אין ניתוקבudp) נקבל פרמטר נוסף -t המתאר timeout. לאחר שיחלפו timeout שניות יש להרוג את כל ההליכים.
הערה כדי לתמוך בtimeout - מומלץ לקרוא ולהשתמש בפונקציה alarm(2). במידה ולא התקבל פרמטר -t עצור את השרת רק אם ה-executable מסתיים. (ולא משום סיבה אחרת).

Examples:

mync -e “ttt 123456789”-i UDPS4050 -t 10
יפתח שרת UDP בפורט 4050 ויאזין לקלט משם. הפלט עדין יוצא לstdout. הרוג את השרת אחרי 10 שניות

mync -e “ttt 123456789” -i UDPS4050 -o TCPClocalhost,4455
יפתח שרת UDP בפורט 4050 ויאזין לקלט משם. הפלט יוצא כTCPClient לport 4455.

שים לב אפשר לערבב תקשורת TCP ו UDP.

תרגיל 5 - mynetcat - שלב 4 (30 נקודות)

לא מומלץ למי שלא פתר כבר (תרגיל קשה ועופר הוריד אותו)

שכללו את הפרמטרים -b, -o, i לתמוך גם בפרמטרים הבאים

TCPMUXS<PORT>
יפתח שרת TCP תומך IO MUX בPORT.
מותר להשתמש ב$select(2)$, $poll(2)$ או בכל דרך אחרת לבחירתכם.

במידה ויש יותר מלקוח אחד שמתחבר תפתחנה הרצות נוספות של התוכנה שהתקבלה ב-e לתמיכה בלקוחות הנוספים.
לדוגמא :

mync -e “ttt 123456789” -b TCPMUXS4050

התחל שרת TCP, IO MUX בport 4050.

שלב 6 - שימוש ב-Unix domain sockets לתקשורת - 30 נקודות

שכלל את mync לתמוך גם ב-unix domain sockets

הוסף את הפרמטרים הבאים

UDSSD<path>
Unix domain sockets - server - datagram open server on path
לתמוך רק בקלט - כלומר -i

UDSCD<path>
Unix domain sockets - client - datagram connect to path.
לתמוך רק בפלט - כלומר -o

UDSSS<path>
Unix domain sockets - server - stream open server on path
UDSCS<path>
Unix domain sockets - client - stream connect to path.

שימו לב - אין דרישה לתמוך בIO MUX בunix domain sockets.

הנחיות נוספות

1. מומלץ להשתמש ב-$getopt(3)$ לטיפול בפרמטרים.
2. מומלץ להשתמש ב-$alarm(2)$ למימוש timeout
3. ב7.10 ישראל סבלה מתקפה מהקשות בתולדותיה. עקב כך ועקב העומס הנפשי הנגרם לסטודנטים במלחמה כל חלקי שלב 1 הקשורים לlunar deportation ירדו מהתרגיל.
4. יש לצרף לתרגיל דוח (או דו"חות) code coverage.
5. יש לצרף makefile רקורסיבי הבונה את כל השלבים
6. שים לב משקלו של תרגיל זה הוא 10% מהציון הסופי ו5% הגנה אנא התייחסו בהתאם.
7. בגרסה 1.1 של העבודה התווסף שלב 2.5 והוגדל סך הניקוד ל-130 נקודות. הדבר מאפשר לכם לבחור לוותר על מימוש שלב 5 או שלב 6 ועדיין להגיע לציון מלא. אפשר עדיין לממש את שני הסעיפים ולקבל עבורם ניקוד, אבל הציון בכל מקרה לא יעבור את 100 הנקודות.

שינויים - מעקב אחרי גירסאות

1.1

מספר הבהרות והקלות - עופר

1.2

ביטלתי את היכולת להוציא פלט אם אנחנו שרת udp
ביטלתי את היכולת לקבל קלט אם אנחנו לקוח udp

1.3

ביטלתי את היכולת להוציא פלט אם אנחנו שרת unix domain socket
ביטלתי את היכולת להוציא קלט אם אנחנו לקוח unix domain socket