מבוא לתכנות מערכות

מידע כללי לגבי תרגילי הבית

1.1 אופן מתן הציון

הקוד שתגישו בתרגילים יעבור על ידינו בדיקה אוטומטית, ולכן על התוכנית שלכם ליצר בדיוק את הפלט הנדרש בתרגיל.

תכנית הבדיקה האוטומטית מפנה את פלט ה-stdout ופלט ה-stderr אשר התכנית שלכם מייצרת לתוך קבצים שונים, ואח"כ מפעילה את הפקודה diff של UNIX, כדי לוודא שתוכן הקבצים זהה בדיוק למצופה. לפיכך הבדל של תו אחד (אפילו סוף שורה או רווח מיותר) בין הקובץ המצופה לבין הקובץ שהתוכנית שלכם תיצור יגרום לכך שתכניתכם תיכשל בבדיקה שלנו. בכדי למנוע פגיעה מיותרת בציון, הקפידו להוציא פלט בדיוק לפי ההגדרות. על מנת שתוכלו לבדוק את עצמכם, מסופקים לכם בדיקות ("טסטים") וכן סקריפט שיאפשר לכם לבדוק את תקינות הקובץ שאתם מתכוונים להגיש.

1.2 החלפת ה-shell

ניתן להשתמש בפקודה הבאה כדי לקבל את שם ה-Shell הנוכחי ב-Unix

```
> echo $0
```

במחשבים בהם Bash אינה Shell ברירת המחדל (כמו בשרת ה-t2 לדוגמה), ניתן להשתמש בפקודה chsh (change shell) כדי לשנות את ברירת המחדל, על פי ההוראות הבאות. מה שמודגש באדום צריך להקליד. חשוב להקפיד לכתוב את שמו המוחלט הנכון של קובץ ההרצה של ה-Shell החדש, אחרת החשבון ייהפך ללא שמיש!!

```
>chsh
Changing shell for mtm.
old shell: /bin/tcsh
New shell: /bin/bash
Shell will be changed for mtm in approximately 5 minutes
>
```

במידה והחלטתם להשאר עם ה Shell הקיים (בדרך כלל tcsh) תוכלו תמיד לעבור ל Shell אחר כמו bash פשוט על ידי הרצתו. בכל סקריפט, ניתן לציין בשורה הראשונה את ה-Shell אשר ישמש להרצת הסקריפט (ואף רצוי, אחרת ה-Shell אשר יריץ את הסקריפט תלוי ב-Shell אשר ממנו הסקריפט יופעל).

1.3 ייצוג ה-newline

כפי שלמדנו בתרגולים, קיימות מערכות הפעלה נוספות חוץ מ- Windows, וכמובן שיש ביניהן הבדלים. אחד ההבדלים הוא התווים שמייצגים את סוף השורה בקובץ טקסט (newline). בכל מערכת הפעלה שהיא דמוית יוניקס (Unix-like), ובפרט בלינוקס, התיו הבודד '\n' (שנקרא "Line Feed") מייצג newline (כלומר בסוף כל שורה יש תיו '\n'). בעוד שווינדוס, מסיבה היסטורית משעשעת[1], ה-newline מיוצג ע"י רצף של שני תווים: '\r\n'.

לכן כשתעבירו קבצים מ- Windows ללינוקס, עלולות להיווצר בעיות (סקריפטים עם סינטקס לא חוקי, קבצי קלט שלא מפורשים כראוי, קבצים שאמורים להיות זהים – אבל diff צועק שהם לא, למרות ש "diff –w" לא יצעק, וכו'). למזלכם קיימת תוכנית מאוד פשוטה בלינוקס שממירה את ה- newlines בקובץ נתון, מהפורמט של Windows ('\r\n') לפורמט של יוניקס ('\n'). התוכנית נקראת dos2unix, והיא מקבלת רשימה של קבצים שאותם היא משנה. דוגמת הרצה:

```
dos2unix a.txt b.txt
```

1.4 מימוש הפתרון

- שימו לב שבכל תרגילי C/C++ אותם תגישו (תרגילים 2-4), על המימוש לעבוד ללא שגיאות זכרון (גישות לא חוקיות וכדומה) וללא דליפות זכרון. ניתן ומומלץ להשתמש ב- vlagrind על מנת לאתר דליפות זיכרון. valgrind הוא כלי המותקן על ה- t2 ומשרת בין היתר למטרה זו.
- יש לכתוב את הקוד בפונקציות קצרות וברורות. במידה והנכם נדרשים לבצע פעולה מסובכת, חישבו כיצד ניתן לפרק אותה לתת פעולות.

1.5 הגשה אוטומטית

- ע"מ לבטח את עצמכם כנגד תקלות בהגשה האוטומטית, שמרו את קוד האישור עבור ההגשה.
- שימו לב, ניתן להגיש את התרגיל מס' פעמים. ההגשה האחרונה היא הנחשבת.

1.6 דגלים

בתרגילי C/C++ (תרגילים 2-4), הקוד יקומפל עם מספר דגלים שכדאי להכיר:

- -std=c99 : קביעת C99 כתקן לשפה (עבור תרגילי C). C99 הוא תקן חדש יותר מ- ANSI C המקורי, המאפשר בין היתר להצהיר על משתנים באמצע שגרות, להשתמש ב- bool ועוד.
- -Wall : דיווח על כל האזהרות

---

[1] ב-windows מייצגים newline ע"י שני תוים אשר מייצגים את שתי הפעולות המכניות שמכונת כתיבה (!!!) צריכה לעשות בסיום שורה: החזרת המסילה שמאלה/ימינה לנקודה ההתחלתית שלה ("Carriage Return" שמיוצג ע"י '\r') והזזת דף הנייר בשורה אחת למעלה ("Line Feed" שמיוצג ע"י '\n').

- pedantic-errors- : דיווח על סגנון קוד שאינו עומד בתקן הנבחר כשגיאות.
- Werror- : התייחס לאזהרות כאל שגיאות, משמעות דגל זה הוא שהקוד חייב לעבור הידור ללא אזהרות.
- DNDEBUG- : מוסיף את השורה #define NDEBUG בתחילת כל יחידת קומפילציה. בפועל, דלג זה יגרום לכך שהמאקרו assert (אם בשימוש) לא יפריע ולא יופעל בריצת התכנית.

1.7 שונות

טיפ כללי לחיים: כדאי לכם מאוד להכיר את Notepad++ (ל-Windows בלבד) - עורך טקסט קליל ומאוד שימושי, שבין היתר מאפשר לכם לראות את תווי ה-newline, ואף לבצע את ההמרה שהזכרנו קודם.