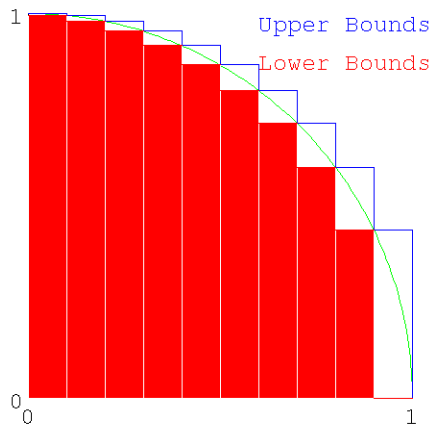
系統程式期中考（上機考部分）

計分方式，請先從 1.I 及 2.I 選擇一題作為你的主答，主答為 60 分，之後每一題佔 10 分，最高 100 分。

1. Ron 博士覺得蒙地卡羅方法使用了 random 函數，而 random 函數式系統的效能瓶頸所在，因此他想改用數值積分方法計算 pi，數值積分方法是在 X 座標上分成 n 個等間距的點，接下來使用這 n（介於 0~1，包含 0 及 1）個點產生 n-1 個矩形去逼近 pi 的值。矩形的畫法有二種，第一種採取左邊的線與圓的交叉點為高，畫出矩形（如圖中藍色的部分）。第二種採用矩形右邊的線與圓的交叉點為高，畫出矩形（如圖中紅色的部分）。由於藍色的部分的面積一定大於 1/4 圓，因此是上界（即圖中的 upper bounds），而紅色的部分的面積一定小於 1/4 圓，因此是下界（lower bounds），透過上、下界我們就可以知道算出來的 pi 的準確度到小數點底下第幾位。
   I. （60pt/10pt）執行檔名稱為 pi，執行方式為 pi ###### $$，其中######為 X 軸上的等間距點的數量，$$ 為執行緒的數量。請輸出 pi 的上界及下界。（hint，上界和下界的矩形「剛好差一格」，你可以藉此來加速運算。換句話說下界的部分向右平移一個，再算上界的的最左邊一個，就可以得到上界）
   II. （10pt）令######一定是 2 的冪次方（例如：4096），以漸進的方式計算 pi，例如：第一回合是 8 個點，第二回合是 16 個點。每當使用者按下 ctr-c 時印出當時已經計算出來的 pi 的值的上界及下界。使用者連續按下 ctr-c（在 2 秒內按下二次）時終止程式並印出 pi 的值的上界及下界。請注意，使用漸進方式求 pi 時，你必須使用上一個回合的結果繼續往下推算。
   III. （10pt）執行 pi 程式時可以下達準確性參數，指令形式是 pi -pre 15，15 表示 pi 必須準確到小數點底下第 15 個位數，請使用漸進的方式的算法計算 pi，當使用者按下 ctr-c 時印出當時已經計算出來的 pi 的值的上界及下界。使用者連續按下 ctr-c（在 2 秒內按下二次）時終止程式並印出 pi 的值的上界及下界。請注意，使用漸進方式求 pi 時，你必須使用上一個回合的結果繼續往下推算。
   IV. 繳交：pi.c、makefile、readme.pi，在 readme.pi 中說明你完成了哪些功能，約略說明如何完成這些功能
   V. 協助你判斷程式的準確性，pi 的近似值為：
   3.14159265358979323846264338327950288419716939937510582097494459230781640628



2. Ron 博士想要擴充 myshell.c 的功能，請依照下列的要求修改 myshell.c

   I. （60pt/10pt）當使用者按下 ctr+\的時候，如果 myshell 正命令一個 child 在執行命令，那麼將 ctr+\送給這個 child，如果 myshell 並沒有叫 child 做任何事情，忽略 ctr+\。（hint：ctr+\的 signal 代號是 SIGQUIT）

   II. （10pt）Ron 博士希望 myshell.c 支援 FIFO 的功能，請支援下面的命令

      i. mkfifo myfifo

      ii. ls > myfifo &　，執行 ls 指令，並且將 ls 的 stdout 接到 myfifo，由於這道指令最後面是&字元，因此請在背景執行 ls > myfifo

      iii. sort < myfifo　，執行 sort 指令，並且將 sort 的 stdin 接到 myfifo

      （hint：mkfifo 是 Linux 的工具程式，因此你應該不需要特別的實作，上述命令都能在 bash 中使用，因此你可以先用 bash 觀察指令的行為）

   III. 繳交：myshell.c、makefile、readme.fifo，在 readme.fifo 中說明你完成了哪些功能，約略說明如何完成這些功能