ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

C programming examples

Prof. Gustavo Alonso
Computer Science Department
ETH Zürich
alonso@inf.ethz.ch
http://www.inf.ethz.ch/department/IS/iks/

Programming examples in C

- Dynamic queue
- Use of timing functions
- Generating random numbers
- Arguments and main
- Use of static
- Counting bits

Programming examples taken from the on-line book:

Programming in C

UNIX System Calls and Subroutines using C

http://www.cs.cf.ac.uk/Dave/C/CE.html

```
 /*                                                    */
/* queue.c                                             */
/* Demo of dynamic data structures in C                */

#include <stdio.h>

#define FALSE 0
#define NULL 0

typedef struct {
    int     dataitem;
    struct listelement *link;
}               listelement;

void Menu (int *choice);
listelement * AddItem (listelement * listpointer, int data);
listelement * RemoveItem (listelement * listpointer);
void PrintQueue (listelement * listpointer);
void ClearQueue (listelement * listpointer);

main () {
    listelement listmember, *listpointer;
    int     data, choice;
    listpointer = NULL;
```

```
Menu (&choice);
switch (choice) {
   case 1:
      printf ("Enter data item value to ad
      scanf ("%d", &data);
      listpointer = AddItem (listpointer, d
      break;
   case 2:
      if (listpointer == NULL)
         printf ("Queue empty!\n");
      else
         listpointer = RemoveItem (listpo
      break;
   case 3:
      PrintQueue (listpointer); break;
   case 4:
      break;
   default:
      printf ("Invalid menu choice - try ag
      break;

ile (choice != 4);
rQueue (listpointer);
(0);
                    /* main */
```

```
void Menu (int *choice) {

        char    local;

        printf ("\nEnter\t1 to add item,\n\t2 to remove item\n\
    \t3 to print queue\n\t4 to quit\n");
        do {
            local = getchar ();
            if ((isdigit (local) == FALSE) && (local != '\n')) {
                printf ("\nyou must enter an integer.\n");
                printf ("Enter 1 to add, 2 to remove, 3 to print, 4 to quit\n");
            }
        } while (isdigit ((unsigned char) local) == FALSE);
        *choice = (int) local - '0';
    }
```

```
listelement * AddItem (listelement * listpointer, int data) {

    listelement * lp = listpointer;

    if (listpointer != NULL) {
        while (listpointer -> link != NULL)
            listpointer = listpointer -> link;
        listpointer -> link = (struct listelement  *) malloc (sizeof (listelement));
        listpointer = listpointer -> link;
        listpointer -> link = NULL;
        listpointer -> dataitem = data;
        return lp;
    }
    else {
        listpointer = (struct listelement  *) malloc (sizeof (listelement));
        listpointer -> link = NULL;
        listpointer -> dataitem = data;
        return listpointer;
    }
}
```

```
listelement * RemoveItem (listelement * listpointer) {

    listelement * tempp;
    printf ("Element removed is %d\n", listpointer -> dataitem);
    tempp = listpointer -> link;
    free (listpointer);
    return tempp;
}

void PrintQueue (listelement * listpointer) {

    if (listpointer == NULL)
        printf ("queue is empty!\n");
    else
        while (listpointer != NULL) {
            printf ("%d\t", listpointer -> dataitem);
            listpointer = listpointer -> link;
        }
    printf ("\n");
}
```

```
void ClearQueue (listelement * listpointer

    while (listpointer != NULL) {
        listpointer = RemoveItem (listpointer
    }
}
```

Use of timing functions

```
/* timer.c */
/* Computes the time in seconds to do a computation */

#include <stdio.h>
#include <sys/types.h>
#include <time.h>

main()
 {                          int i;
                            time_t t1,t2;

                            (void) time(&t1);
                            for (i=1;i<=300;++i)
                              printf(``%d %d %dn",i, i*i, i*i*i);
                            (void) time(&t2);
                            printf(``n Time to do 300 squares and
                           cubes= %d secondsn", (int) t2-t1);
                  }
```

Generating random numbers

```
/* random.c */
        #include <stdio.h>
        #include <sys/types.h>
        #include <time.h>

        main()
          {             int i;
                         time_t t1;

                        (void) time(&t1);
                         srand48((long) t1);
                         /* use time in seconds to set seed */
                         printf("5 random numbers (Seed = %d):n"",(int) t1);
                         for (i=0;i<5;++i) printf("%d "", lrand48());
                         printf("\n\n""); /* flush print buffer */
                         }
```

On random number generation in C

- lrand48() returns non-negative long integers uniformly distributed over the interval (0, 2**31).
- A similar function drand48() returns double precision numbers in the range [0.0,1.0).
- srand48() sets the seed for these random number generators. It is important to have different seeds when we call the functions otherwise the same set of pseudo-random numbers will generated. time() always provides a unique seed.

Arguments and main

```
#include <stdio.h>

main(int argc, char **argv)

{ /* program to print arguments from command line */

    int i;

    printf("argc = %d\n\n",argc);
    for (i=0;i<argc;++i)
                printf("argv[%d]: %s\n",i, argv[i]);
}
```

Use of static

```
#include <stdio.h>

void stat();

main() {
    int counter;    /* loop counter */

    for (counter = 0; counter < 5; counter++) {
        stat();
    }
}

void stat()
{ int temporary = 1;
  static int permanent = 1;
        (void)printf("Temporary %d Permanent %d\n",
            temporary, permanent);
        temporary++;
        permanent++;
}
```

Counting bits

```
int bitcount(unsigned char x)

        { int count;

                for (count=0; x != 0; x>>=1)
                        if ( x & 01)  count++;
                return count;
        }
```