Universität Innsbruck - Institut für Informatik
Prof. Clemens Ballarin, Robert Binna, Friedrich Neurauter, Francois Scharffe and Sarah Winkler

22. April 2008

Proseminar Algorithmen und Datenstrukturen

Exercise sheet 5

Exercise 1 (Records)

This exercise deals with representation of records in C.

a) Consider the following C program:

```
#include <stdio.h>

struct example_struct{ char c1; char c2; } s;

int main() {

 printf("size of s: %d\n",sizeof(s));
 printf("size of s elements: %d\n", sizeof(s.c1)+sizeof(s.c2));

 return 0;
}
```

Compile and run it. Now consider the following program:

```
#include <stdio.h>

struct example_struct{ char c1; char c2; int i;} s;

int main() {

 printf("size of s: %d\n",sizeof(s));
 printf("size of s elements: %d\n", sizeof(s.i)+sizeof(s.c1)+sizeof(s.c2));
```

```
 return 0;
}
```

Consider now the following program:

```
#include <stdio.h>

struct example_struct{ char c; int i;} s;

int main() {

 printf("size of s: %d\n",sizeof(s));
 printf("size of s elements: %d\n", sizeof(s.i)+sizeof(s.c));

 return 0;
}
```

And finally this one:

```
#include <stdio.h>

struct example_struct{ char c1; int i; char c2;} s;

int main() {

 printf("size of s: %d\n",sizeof(s));
 printf("size of s elements: %d\n", sizeof(s.i)+sizeof(s.c1)+sizeof(s.c2));

 return 0;
}
```

What happens ? Explain the results and give a schema representing what happens in the memory stack for each of these examples.

b) Write a program that stores the names, two line addresses and ages of a group of people. Each person should be stored in a structure. Use an array of such structures to hold the data for a group of people. Write functions for reading a person's details from the standard input stream and for printing a person.

c) Consider the following record (in Pseudo-code):

Listing 1 RECORDS

```
1: record T =
2: begin
3:    n1 : character;
4:    n2 : integer;
5: end
6: A is declared as: array [1..10] of array [1..10] of T;
7: B is declared as: array [1..5] of array [1..5] of T;
```

Given the size of a character is 1 and the size of an integer is 4.

- What's the size of A?
- What's the position of $A[3][5].n1$ in memory relative to the base address of A?
- What's the size of B?
- What's the position of $B[2][4].n2$ in memory relative to the base address of B?

Exercise 2 (Data structures)

What are the data structures needed to construct:

- a doubly-linked list ?
- a tree with branching factor n ?
- a tree with arbitrary branching factor ?

Give these structures both in pseudo-code and in C code.

Exercise 3 (Linked List)

In this exercise you will implement a linked list: Provide insert, delete, search and print methods to manipulate the list according to the following specifications:

a) **data** the data in each cell is an integer.

add this method adds a given integer value at the list head.

delete this method removes a given integer value from the list.

search this method searches the list for a given integer value. It returns TRUE in case of success, FALSE otherwise.

print this method prints the list on the standard output.

Provide a main containing an example of creating a list, printing it and deleting some of its values.

b) **append** this method appends two lists given as arguments such that the first cell of the second list is linked to the last cell of the first list.

insert this method inserts a given integer value into the list such that the list is always ordered with increasing values.

Update your main to demonstrate these functions.