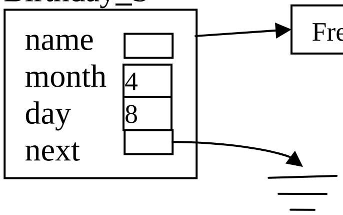
01Lecture notes for May 1, 2005

Transcribed by Mike Parker, edited by P. Conrad

Method 1 (analogy: pick-up truck pulling a camper)

Birthday_S

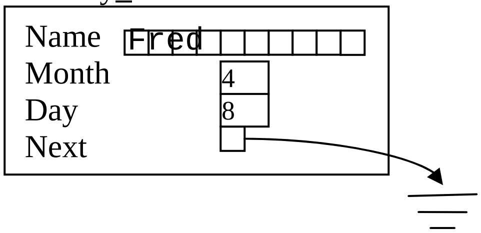


```
struct Birthday_S
{
   char name;  // ptr to space on heap
   int month;
   int day;
   Birthday_S *next;
};
```

Vs.

Method 2 (analogy: an RV)

Birthday_S



```
const int MAX_NAME_LEN = 10;
struct Birthday_S
{
   char name[MAX_NAME_LEN];
   int month;
```

```
  int day;
  birthday_S *next;
};
```

Needed for both methods

```
Birthday_S, *head, *tail; // declaring
head = tail = NULL; // or head = (tail = NULL) because = is right associative
infile.getline(inputLine, 1024);
while (!infile.eof(0))
{
   // process the input line
   char *namePtr = strtok (inputLine, NULL, “,”)
   char *monthPtr = strtok (inputLine, NULL, “,”)
   char *dayPtr = strtok (inputLine, NULL, “,”)
   // process 3 ptr’s
   // create new node
   // add new node to link list
   infile.getline (inputLine, 1024);
}
```

Method 1

```
Birthday_S *p;
p = newBirthday_S;
p -> name = new char [strlen(namePtr) + 1]; // allocates new space on heap
strcpy (p -> name, namePtr); // puts Fred in name
p -> month = atoi (monthPtr);
p -> day = atoi (dayPtr); // for double use ‘atof’
p -> next = NULL; // initially set to NULL
```

Method 2

```
Birthday_S *p;
p = new Birthday_S;
strncpy (p -> name, namePtr, MAX_NAME_LEN);
p -> name [MAX_NAME_LEN – 1] = ‘\0’; // or 0, or NULL
```

.

.

. rest is the same

.

.

Basic code to add a node to a linked list

```
if (head = NULL)
   {
      head = p;
   }
else
   {
      tail -> next = p;
   }
tail = p;
```

A function to add a node to a linked list

If we make that into a function, the pointers become “double stars” because we are passing a pointer by reference (the old “C-style” pass by reference).

```
void addToList (Birthday_S **headPtr, Birthday_S **tailPtr, Birthday_S *p)
{
   if ((*headPtr) = NULL)
     {
      (*headPtr) = p;
     }
   else
     {
      (*tailPtr) -> next = p;
     }
   (*tailPtr) = p;
}

// need ** because head and tail are local in addToList
```