COMP1921

Function Pointers in C
Function Pointers

- C can pass functions by passing a pointer to them.
- Function pointers ...
  
  - are references to memory addresses of functions
  
  - are pointer values and can be assigned/passed

- E.g. a pointer to a function mapping

  \[
  \text{int} \rightarrow \text{int} \\
  \text{int} (*\text{fun})(\text{int})
  \]

- Function pointer variables/parameters are declared as:

  \[
  \text{typeOfReturnValue} (*\text{fname})(\text{typeOfArguments})
  \]
Function Pointers

int square(int x) { return x*x;}

int timesTwo(int x) {return x*2;}

int (*fp)(int);

fp = &square;  //fp points to the square function

int n = (*fp)(10);  //call the square function with input 10

fp = timesTwo;  //works without the &
    //fp points to the timesTwo function

n = (*fp)(2);  //call the timesTwo function with input 2

n = fp(2);  //can also use normal function call
    //notation
Higher-order Functions

• Functions that get other functions as arguments, or return functions as a result

• **Example:** the function `traverse` takes a list and a function pointer as argument and applies the function to all nodes in the list

```c
void printList(link ls){
    link curr = ls;
    while(curr != NULL){
        printf("%d ",curr->data);  //Process the node
        curr = curr->next;
    }
}

void traverse (link ls, void (*fp) (link)){
    link curr = ls;
    while(curr != NULL){
        (*fp) (curr);
        curr = curr->next;
    }
}

// To call the function, function must have matching prototype
traverse(myList, printList);
```

Second parameter is `fp`, Pointer to a function like,
`void functionName(link ls)`

First parameter is `ls`, of type `link`

// apply function fp to all nodes in ls

Higher-order Functions

void printNode(link ls){
    if(ls != NULL){
        printf("%d->", ls->data);
    }
}

void printGrade(link ls){
    if(ls != NULL){
        if(ls->data>= 85){
            printf("HD ");
        }
        else {
            printf("FL ");
        }
    }
}

void traverse (link ls, void (*fp) (link));

//To call the function
//Function must have matching prototype

traverse(myList, printNode);
traverse(myList, printGrade);