```
#include <stdlib.h>
#include <stdio.h>
#include <setjmp.h>

#include "exception.h"
#include "memory.h"

DEFINE_EXCEPTION(any);

struct exception
{
    struct exception *header;
};

struct context
{
    jmp_buf context_buffer;
    struct context *previous;
    struct exception *header;
};

/* This flag is used to indicate exception caught */
unsigned char exception_flag = 0;

/* Here is linked the stack of contexts */
static struct context *top = NULL;

/* This hook refers the function used to emit an error message on
   uncaught exception */

static void
default_send_uncaught_message(char *name, char *file, int line)
{
    fprintf(stderr,
            "Uncaught exception <%s> raised from \"%s:%d\"\n",
            name, file, line);
}
```

```
static void (*send_uncaught_message)(char *name, char *file, int line)
    = default_send_uncaught_message;

int
exception_raised(void)
{
    return exception_flag != 0;
}

void *
exception_jmp_buf(exception e)
{
    if (top == NULL)
        return NULL;
    return top->context_buffer;
}

void
exception_uncaught(exception e, char *name, char *file, int line)
{
    send_uncaught_message(name, file, line);
    exit(EXIT_FAILURE);
}

void
exception_push(exception e)
{
    struct context *c = memory_alloc(sizeof(struct context));

    c->previous = top;
    c->header = e;
    top = c;
}
```

```
int
exception_search(exception e)
{
    for (;;)
    {
        if (top == NULL)
            return 0;
        if (top->header == e || top->header == EXCEPTION_NAME(any))
            return 1;
        exception_pop(e);
    }
}

void
exception_pop(exception e)
{
    struct context *c = top->previous;

    memory_free(top);
    top = c;
}
```