**Отчет по лабораторной работе № 7**

Тарасов А. Т.

Группа ИУ9-31Б

**Задача:** Реализовать http прокси сервер

**Исходный код программы:**

package main

import (

    "fmt"

    "io"

    "log"

    "net/http"

)

type HttpConnection struct {

    Request  \*http.Request

    Response \*http.Response

}

type HttpConnectionChannel chan \*HttpConnection

var connChannel = make(HttpConnectionChannel)

//Вывод заголовков запроса и ответа

func PrintHTTP(conn \*HttpConnection) {

    fmt.Printf("%v %v\n", conn.Request.Method, conn.Request.RequestURI)

    for k, v := range conn.Request.Header {

        fmt.Println(k, ":", v)

    }

    fmt.Println("==============================")

    fmt.Printf("HTTP/1.1 %v\n", conn.Response.Status)

    for k, v := range conn.Response.Header {

        fmt.Println(k, ":", v)

    }

    fmt.Println(conn.Response.Body)

    fmt.Println("==============================")

}

func HandleHTTP() {

    for {

        select {

        case conn := <-connChannel:

            PrintHTTP(conn)

        }

    }

}

type Proxy struct {

}

func NewProxy() \*Proxy { return &Proxy{} }

//Обработчик запросов

func (p \*Proxy) ServeHTTP(wr http.ResponseWriter, r \*http.Request) {

    var resp \*http.Response

    var req \*http.Request

    var err error

    client := &http.Client{}

    req, err = http.NewRequest(r.Method, r.RequestURI, r.Body)

    for name, value := range r.Header {

        req.Header.Set(name, value[0])

    }

    resp, err = client.Do(req)

    r.Body.Close()

    if err != nil {

        http.Error(wr, err.Error(), http.StatusInternalServerError)

        return

    }

    conn := &HttpConnection{r, resp}

    for k, v := range resp.Header {

        wr.Header().Set(k, v[0])

    }

    wr.WriteHeader(resp.StatusCode)

    io.Copy(wr, resp.Body)

    resp.Body.Close()

    PrintHTTP(conn)

}

func main() {

    port := "8080"

    proxy := NewProxy()

    fmt.Println("==============================")

    fmt.Println("Starting http proxy on " + port)

    fmt.Println("==============================")

    err := http.ListenAndServe(":" + port, proxy)

    if err != nil {

        log.Fatal("ListenAndServe: ", err.Error())

    }

}

