<table>
<tr><td colspan="3">Politechnika Świętokrzyska w Kielcach
Wydział Elektrotechniki, Automatyki i Informatyki</td></tr>
<tr><td colspan="3">PODSTAWY PROGRAMOWANIA 2– PROJEKT
Informatyka - I rok, Rok akademicki - 2021/2022, semestr - II</td></tr>
<tr><td>I termin</td><td colspan="2">Temat projektu: Menager plików</td></tr>
<tr><td>Grupa: 1ID14B</td><td rowspan="2">Wykonujący:
Marek Supierz,
Andrzej Mysior,
Adrian Nowak</td><td rowspan="2">Ocena:</td></tr>
<tr><td>Data oddania sprawozdania:
05.06.2022</td></tr>
</table>

Prace przewidziane na I termin:

1. Dodanie ewentualnych modyfikacji,
2. Stworzenie dokumentacji w programie Doxygen,
3. Sporządzenie sprawozdania

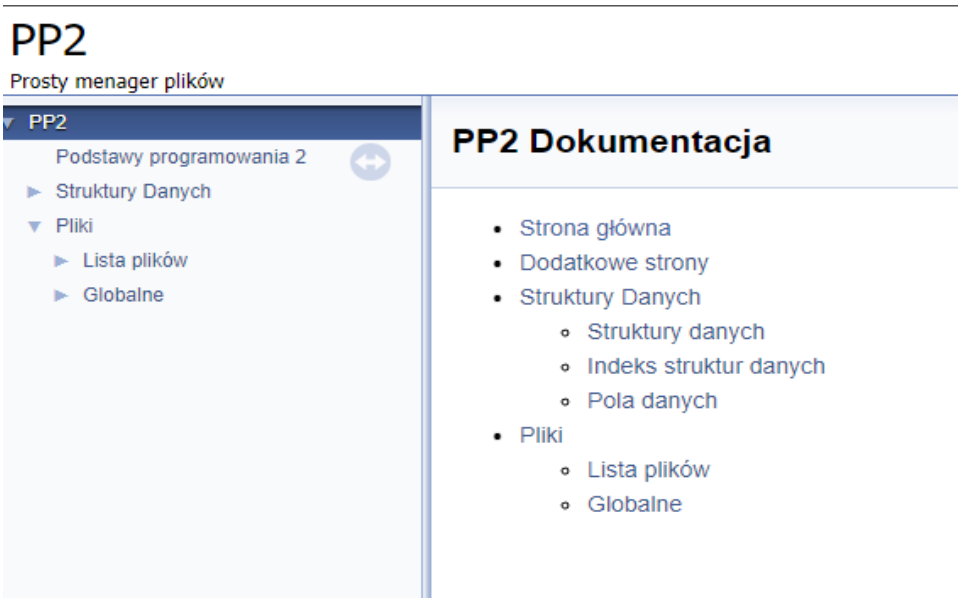
Github: https://github.com/PSK-projekty/PP2
Trello: https://trello.com/b/xM0rMFLY/pp2

Dodanie ewentualnych modyfikacji

Wprowadzono nieznaczne modyfikacje mające na celu poprawę czytelności i przejrzystości a także naprawiono działanie funkcji przenoszącej pliki/katalogi. Podjęto także próbę implementacji biblioteki Curses, lecz i tym razem bez skutku.

Stworzenie dokumentacji w programie Doxygen

Do kodu programu dołączono komentarze a następnie stworzono dokumentację w programie Doxygen.



Podsumowanie prac przewidzianych na I termin

Naprawiono działanie funkcji przenoszącej pliki/katalogi. Błędne działanie było spowodowane pomyłką programisty, w wywołaniu funkcji została podana nieodpowiednia nazwa zmiennej. Spełniono założenia etapu oraz projektu.

Podsumowanie projektu

Opis tematyki projektu

Tematem projektu był prosty menager plików pozwalający wykonywać takie operacje jak:

- Tworzenie/usuwanie katalogów
- Wyświetlanie informacji o plikach
- Zmiana nazwy
- Kopiowanie plików/katalogów
- Wyszukiwanie plików/katalogów
- Zmiana atrybutów
- Przenoszenie plików pomiędzy katalogami

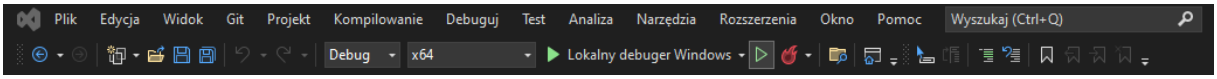
Menager plików to program komputerowy który zapewnia interfejs użytkownika do zarządzania plikami i folderami . Najczęstsze operacje wykonywane na plikach lub grupach plików to tworzenie, otwieranie (np . przeglądanie , odtwarzanie, edytowanie lub drukowanie), zmiana nazwy, kopiowanie, przenoszenie , usuwanie i wyszukiwanie plików, a także modyfikowanie atrybutów plików , właściwości i uprawnień do plików . Foldery i pliki mogą być wyświetlane w drzewie hierarchicznym na podstawie ich struktury katalogów.
Najbardziej znanymi menagerami plików są: Norton commander, Midnight Commander, Dos Navigator, Nautilius.

Użyty język, biblioteki, OS, IDE

Jedynym językiem użytym w projekcie jest język C.
Użyto bibliotek: stdio.h, stdlib.h, string.h, windows.h
Pracowano i testowano pod systemem Windows 10
Posługiwano się środowiskiem programistycznym Visual Studio 2022

Instrukcja kompilacji/uruchomienia, opis działania i instrukcja obsługi

Aby uruchomić program przy użyciu Visual Studio 2022 należy otworzyć plik ‘Projekt PP2.sln’ a następnie uruchomić przy pomocy kombinacji klawiszy Ctrl+F5, lub poprzez naciśnięcie zielonego trójkąta na górnym pasku.

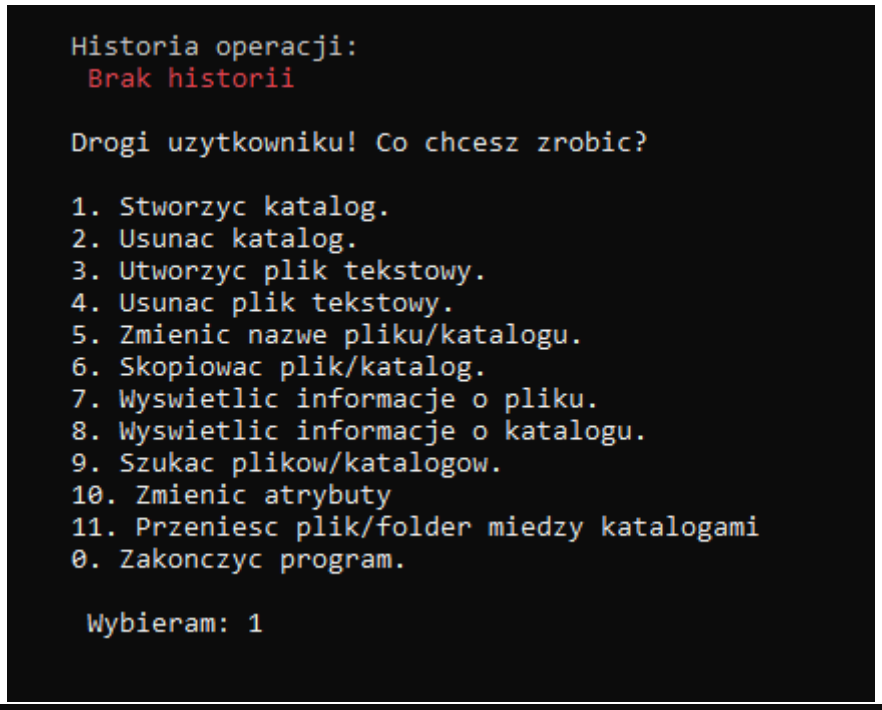


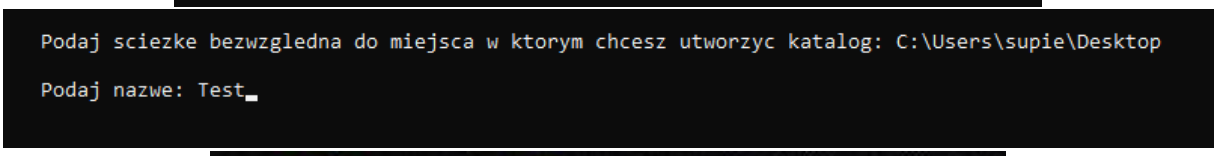
Ze względu na możliwość nieprawidłowego działania niektórych funkcji programu zaleca się jednak użycia środowiska Dev C++.

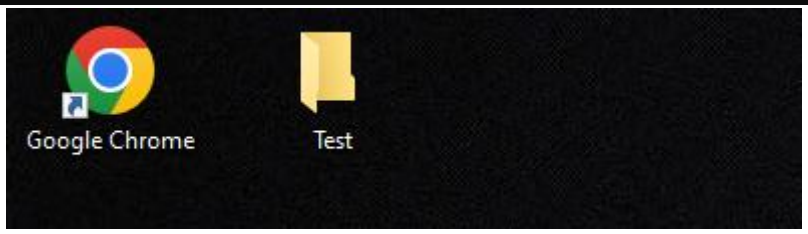
Program działa w oparciu o instrukcję switch która wywołuje odpowiednie funkcje. W nich program prosi o niezbędne dane.

Posługiwanie się programem ogranicza się do wpisania numeru opcji a następnie podania wymaganych danych.

Zrzuty ekranu z przykładowym działaniem







W wybranej lokalizacji został utworzony folder „Test”.

Opis użytych algorytmów i najważniejsze fragmenty implementacji

```
void create_directory(char path[], char name[]) {

    char command[] = "md ";
    char mark[] = "\\";
    char cmd[100] = "";

    strcat(cmd, command);
    strcat(cmd, path);
    strcat(cmd, mark);
    strcat(cmd, name);

    system(cmd);
}
```

Powyższa funkcja odpowiada za tworzenie katalogów. W pierwszej kolejności tworzone są dwie zmienne. Pierwsza przechowuje polecenie systemowe. Do drugiej zmiennej będą dołączane kolejne elementy. Przez metodę 'strcat()' najpierw dołączana jest zmienna 'command' a następnie dane pobrane od użytkownika. Na końcu wywoływana jest funkcja 'system()' ze zmienna 'cmd' jako parametrem.

Pozostałe funkcje obsługujące wymagane funkcjonalności działają na wyżej opisanej zasadzie.

Jednym z wymagań projektowych było wykorzystanie dynamicznej struktury danych, listy.

```
typedef struct node {
    int data;
    struct node* next;
}node;
```

Zdefiniowanie struktury i stworzenie głowy. Każdy element listy przedstawiony będzie za pomocą typu strukturowego z dwoma polami. Pierwszym przechowującym daną i drugim wskazującym na kolejny element listy.

```
void push_back(node** head, int option) {

    if (*head == NULL) {
        *head = (node*)malloc(sizeof(node));
        (*head)->data = option;
        (*head)->next = NULL;
    }

    else {
        node* current = *head;

        while (current->next != NULL)
            current = current->next;

        current->next = (node*)malloc(sizeof(node));
        current->next->data = option;
        current->next->next = NULL;
    }
}
```

Funkcja odpowiada za dodawanie elementów na końcu listy.
Trzeba rozważyć dwa przypadki.

1. Gdy head jest NULLem
2. Gdy head nie jest NULLem

Jeśli pierwszy element listy jest pusty, czyli lista nie istnieje. Weź pierwsza komórkę listy, zarezerwuj tylko tyle pamięci ile potrzebujesz dla zmiennych ze struktury. Następnie do zmiennej 'data' przekaz wartość zmiennej 'option'.Weź pierwszy element listy, stwórz kolejny i ustaw jego wartość na NULL.

W przeciwnym wypadku weź pierwszy element i przypisz do zmiennej 'currnet'. Dopóki kolejne elementy listy nie są puste iteruj po liście. Weź pusty element na końcu listy i zarezerwuj miejsce. Do utworzonego miejsca przypisz zmienna. Na końcu utwórz kolejny element listy i ustaw jego wartość na NULL.

```
void show(node* head) {
    printf("\n");
    if (head == NULL) {
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 12);
        printf("\t Brak historii");
    }
    else {
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 10);
        node* current = head;
        do {
            printf(" %d ", current->data);
            current = current->next;
        } while (current != NULL);
    }
}
```

Funkcja odpowiada za wyświetlanie listy.
Trzeba rozważyć dwa przypadki.

1. Gdy head jest NULLem
2. Gdy head nie jest NULLem

Jeżeli head jest pusty, lista jest pusta. Wyświetli się komunikat "Brak historii".

W innym przypadku weź pierwszy element i przypisz do zmiennej 'current'. Następnie wypisz zawartość zmiennej data. Przejdź do kolejnego elementu listy. Iteruj po liście aż do napotkania NULLa.

Podział prac w zespole

Prace zostały podzielone w następujący sposób:

Marek Supierz: programowanie, przygotowanie i zarządzanie repozytorium, testowanie, poprawa błędów, wsparcie przy komentarzach

Andrzej Mysior: Testowanie, przeniesienie harmonogramu do Trello, wsparcie programistyczne, komentarze, przygotowanie dokumentacji

Adrian Nowak: Testowanie, wyszukiwanie potrzebnych zasobów, wsparcie programistyczne, sporządzanie sprawozdań

Podsumowanie

Udało się stworzyć program który poprawnie wykonuje wszystkie wymagane funkcje. Możliwymi dalszymi pracami było by uproszczenie strony użytkowej, na przykład przez możliwości zawarte w bibliotece PDCurses. Możliwym kierunkiem działania było by również rozbudowanie programu o inne funkcjonalności, takie jak przechodzenie przez kolejne katalogi i wyświetlanie ich zawartości.

Nie udało się stworzyć przyjaźniejszej, prostszej strony użytkowej.