Opis struktury programu

io – moduł obsługujący całe wejście/wyjście.
Zawiera funkcje:

- image scanAlphabet() - wczytuje alfabet z katalogu ./alphabet/ do programu zwracająca obiekt typu image
- void printTypeExpression() - drukuje zapytanie o wyrażenie matematyczne
- void printGettingStarted() - drukuje krótki opis programu po jego uruchomieniu
- bool scanExpression(char *expression) – wczytuje wyrażenie; zwraca false przy błędzie
- void printQuestionForPath() - drukuje zapytanie o ścieżkę do zapisu obrazka
- bool scanpath(char *path) – wczytuje ścieżkę zapisy obrazka; zwraca false przy błędzie
- void printDoneAndQuestionForAgain() - drukuje komunikat o sukcesie wydruku obraka i zapytanie, czy drukować kolejne wyrażenie
- bool scanIfAgain() - sprawdza, czy drukować kolejny obrazek lub zakończyć
- void printImage(image bitmap, const char *path) – drukuje obrazek do pliku i otwiera ten plik

utilities – moduł obsługujący przydatne funkcje niezwiązane konkretnie z innymi modułami
Zawiera definicje:

- STRING_SIZE – maksymalna długość ciągów znaków w programie
- RED, GREEN, YELLOW, MAGENTA, CYAN, WHITE, BLUE, RESET - definicje kolorów używanych przy wypisywaniu funkcją printf()

Zawiera funkcje:

- void appendString(char *s,char c) – konkatenuje ciąg s ze znakiem c
- bool isStringEmpty(char *s) – sprawdzająca, czy ciąg znaków jest pusty
- void clearString(char *s) – ustawia ciąg znaków na pusty
- int maxInt(int a, int b) – zwraca maximum z a,b
- int minInt(int a,int b) – zwraca minimum z a,b
- void clearStdin() – ignoruje strumień wejścia do następnego ‘\n’ lub EOF
- bool doesFileExist(char *filename) – sprawdza czy plik istnieje

tree – moduł obsługujący operacje na wyrażeniach w postaci ciągów znaków infix/ONP oraz drzew
Zawiera definicje:

- OPERATOR – string jest operatorem
- EXPRESSION – string jest wyrażeniem
- RPN_ARRAY_SIZE – maksymalna wielkość tablicy ONP
- RIGHT_ASSOCIATIVITY – łączność w prawo
- LEFT_ASSOCIATIVITY – łącność w lewo
- FUNCTION – operator jest funkcją
- unit – struktura reprezentująca elementarną jednostkę w wyrażeniu tj. operator lub ciąg znaków
  - char expression[STRING_SIZE] – zawartość jednostki
  - bool type – typ jednostki
- node – struktura reprezentująca wierzchołki w drzewie
  - unit value – wartość w wierzchołku
  - node *left, *right – wskaźniki na lewe i prawe dziecko wierzchołka

Zawiera funkcje:

- node *createNode(unit object) – tworzy pusty węzeł
- node *operateOnNode(mode *left, node *right, unit object) – operuje na węzłach
- bool commutative(char operator) – sprawdza, czy operator jest przemienny
- bool isOperator(char character) – sprawdza, czy znak jest operatorem
- int weight(char operator) – zwraca wagę operatora
- int associativity(char operator) – sprawdza, w którą stronę łączy operator

- node *convertExpressionToTree(char *expression) – konwertuje wyrażenie na drzewo, złożenie funkcji convertAlgebraicToRPN i convertRPNToTree
- unit *convertAlgebraicToRPN(char *expression, unsigned int *rpn_size) – konwertuje wyrażenie zapisane infiksowo na ONP
- node *convertRPNToTree(unit *rpn, unsigned int rpn_size) – konwertuje wyrażenie zapisane w ONP na drzewo
- void deleteTree(node *current) – usuwa drzewo i zwalnia pamięć
- void deleteUnitArray(unit *rpn) – usuwa tablicę jednostek i zwalnia pamięć

bitmap – moduł obsługujący operacje na bitmapach
Zawiera definicje:
- BASED – łączenie obrazków jest oparte na połączeniu ich w konkretnym punkcie
- TOP – łączenie horyzontalnie obrazków, doklejając je z góry na dół
- BOTTOM – łączenie horyzontalnie obrazków, doklejając je z dołu do góry
- LEFT – łączenie wertykalnie obrazków, doklejając je z lewej strony
- RIGHT– łączenie wertykalnie obrazków, doklejając je z prawej strony
- MIDDLE – łączenie wertykalnie obrazków, doklejając je na poziomie ich środków
- OTHER– symbol inny niż standardowy operator i litera/cyfra
- SYMBOL – symbol, będący literą/cyfrą
- pixel – struktura reprezentująca piksele
  - unsigned char red – czerwony
  - unsigned char green – zielony
  - unsigned char blue – niebieski
- image – struktura reprezentująca obrazek
  - char magic_number[2] – typ obrazka, w przypadku tego projektu P6
  - unsigned int width – szerokość obrazka
  - unsigned int height – wysokość obrazka
  - unsigned int maxval – maksymalna wartość koloru, w przypadku tego projektu 256
  - pixel *map – tablica pikseli reprezentująca obrazek

Zawiera funkcje:
- image generateBitmapFromTree(image *alphabet, node *head_of_tree) – zwraca obrazek wygenerowany z danego drzewa
- image generateBitmapFromTree(image *alphabet, node *current_node, double scale, unsigned int *baseline_parent) – funkcja rekurencyjna generująca fragmenty obrazka
- image mergeBitmapHorizontal(image left, image right, int spot, unsigned int baseline_left, unsigned int baseline_right) – skleja obrazki horyzontalnie
- image mergeBitmapHorizontal(image left, image right, int spot) – skleja obrazki wertykalnie
- image mergeBitmapAndFreeMemory(image left, image right, double scale, char operation, unsgined int baseline_left, unsigned int baseline_right, unsigned int child_height)– funkcja sklejająca obrazki na różne sposoby, składa inne funkcje sklejające obrazki
- void deleteBitmap(image *bitmap) – usuwa obrazek i zwalnia pamięć
- void setTypeP6(char *magic_number) – ustawia typ obrazka na P6
- image createEmptyImage() - zwraca pusty obrazek
- image createDownscaledImage(image original, double scale) – skaluje obrazek w dół
- image createUpscaledImageVertically(image original, double scale) – skaluje obrazek w górę w osi pionowej
- bool isImageEmpty(image candidate) – sprawdza, czy obrazek jest pusty
- image copyImage(image original) – kopiuje obrazek i zwraca kopię