**PHP ACADEMY**

**CURS PHP**

**METODE DE PROIECTARE A ALGORITMILOR – PROIECTAREA MODULARĂ**

**STUDENT: Moldovai Gianina**

**ORADEA**

**2024**

1. **METODE DE PROIECTARE A ALGORITMILOR – PROIECTAREA MODULARĂ**
   1. **SUBPROGRAMAREA**

Un subprogram este o porțiune de cod care îndeplinește o anumită sarcină și care poate fi apelată (sau invocată) de mai multe ori în cadrul programului principal sau din alte subprograme. Acestea sunt adesea folosite pentru a organiza și structura codul în module mai mici și mai ușor de gestionat. Există două tipuri principale de subprograme, acestea fiind: funcțiile și procedurile.

**Funcții**: sunt subprograme care returnează o valoare. Acestea primesc o serie de argumente, efectuează operațiile necesare și returnează o valoare rezultată. Funcțiile sunt folosite pentru a împărți logica programului în bucăți mai mici, reutilizabile și pentru a obține o valoare calculată.

Exemplu:

function suma($a, $b) {

return $a + $b; }

// Apelarea funcției și afișarea rezultatului

$resultatSuma = suma(5, 3);

echo "Suma este: " . $resultatSuma;

**Proceduri**: sunt subprograme care execută un set de instrucțiuni fără a returna o valoare. Procedurile pot fi folosite pentru a executa anumite acțiuni sau operații, fără a fi necesară returnarea vreunei valori.

Exemplu:

function afisareMesaj($nume) {

echo "Bună ziua, $nume!"; }

// Apelarea procedurii afisareMesaj("John");

Structura subprogramelor poate varia ușor în funcție de limbajul de programare folosit, dar există câteva elemente comune care sunt prezente în majoritatea limbajelor de programare. Iată o structură generală a unui subprogram:

**Semnătura subprogramului**: aceasta include numele subprogramului, lista parametrilor și tipurile de date pe care subprogramul le primește sau le returnează (în cazul funcțiilor). Semnătura este definită înainte de corpul subprogramului și specifică felul în care acesta poate fi apelat.

**Parametri**: Subprogramul poate să primească zero sau mai mulți parametri. Acești parametri sunt variabile folosite în cadrul subprogramului și pot fi folosite pentru a transmite informații către subprogram sau pentru a returna rezultate către programul principal.

**Corpul subprogramului**: Aici este definită logica subprogramului, adică secvența de instrucțiuni care se execută atunci când subprogramul este apelat. Aceste instrucțiuni pot include declarații de variabile, structuri de control (cum ar fi bucle sau instrucțiuni condiționale) și orice alte operații necesare pentru îndeplinirea sarcinii subprogramului.

**Instrucțiuni de returnare** (doar pentru funcții): În cazul funcțiilor, este necesar să fie specificată valoarea care va fi returnată către programul principal. Aceasta se realizează folosind instrucțiunea de returnare, care poate fi plasată oriunde în corpul subprogramului, dar de obicei este utilizată pentru a trimite înapoi rezultatul calculat.

**Apelul subprogramului:** În cadrul programului principal sau în alte subprograme, subprogramul este apelat prin numele său, iar, în cazul funcțiilor, valoarea returnată poate fi stocată într-o variabilă sau folosită în alt mod.

Transmiterea parametrilor în programare implică gestionarea relației dintre parametrii actuali (valori transmise la apelul subprogramului) și parametrii formali (parametrii declarați în semnătura subprogramului). În plus, modul în care valorile sunt transmise între acești parametri poate fi prin valoare sau prin referință/adresă.

**Parametrii actuali și formali**

Parametrii actuali: Acești parametri sunt valorile sau variabilele transmise la apelul subprogramului. Parametrii formali: Acești parametri sunt variabilele declarate în semnătura subprogramului pentru a primi valorile transmise.

**Corespondența între parametrii actuali și formali**

Transmitere prin valoare:

* Parametrii actuali sunt copiați și trimiși subprogramului.
* Orice modificare adusă parametrilor formali în subprogram nu afectează valorile parametrilor actuali.
* Este cea mai simplă și mai sigură metodă de transmitere a parametrilor.

Transmitere prin referință/adresă:

* În loc să se copieze valoarea parametrilor actuali, se transmite o referință la locația lor de memorie (adresa).
* Modificările aduse parametrilor formali în subprogram afectează valorile parametrilor actuali.
* Este utilă atunci când se dorește ca subprogramul să modifice valorile parametrilor actuali.

Transmiterea prin valoare:

* Parametrii sunt transmiși prin valoare la subprogram.
* Modificările aduse parametrilor formali nu afectează valorile parametrilor actuali.
* Util în situațiile în care nu este necesară modificarea valorilor parametrilor actuali.

Transmiterea prin referință/adresă:

* Parametrii sunt transmiși subprogramului prin referință sau adresă (folosind pointeri sau referințe).
* Modificările aduse parametrilor formali afectează direct valorile parametrilor actuali.
* Util în situații în care se dorește modificarea directă a valorilor parametrilor actuali în subprogram.

**Apelul subprogramelor**

Apelul subprogramelor reprezintă o componentă fundamentală a programării moderne, permițând dezvoltatorilor să împartă și să organizeze logic codul în bucăți mai mici și mai ușor de gestionat. În acest referat, vom explora diverse aspecte ale apelului subprogramelor, inclusiv utilizarea lor în cadrul expresiilor, apelul ca instrucțiune și autoapelarea prin recursivitate.

Utilizarea apelului în cadrul expresiilor

Apelul subprogramelor poate fi integrat direct în cadrul expresiilor, permițând programatorilor să utilizeze rezultatul unui subprogram ca parte a unei evaluări mai complexe.

Apel de instrucțiune:

Subprogramul poate fi apelat ca o instrucțiune în sine, fără a fi nevoie să se folosească valoarea returnată. Acest lucru este util atunci când subprogramul efectuează o acțiune sau o operație specifică.

Autoapelare (recursivitate):

Recursivitatea este un concept în care un subprogram se apelează pe sine, direct sau indirect. Această tehnică este utilă în problemele care pot fi împărțite în subprobleme similare mai mici.

* 1. **MODULARIZAREA PROGRAMELOR**

O variabilă este o entitate care poate stoca și reprezenta o valoare sau un set de valori într-un program de calculator. Aceste valori pot varia de la numere și caractere la șiruri de caractere, obiecte sau alte tipuri de date. Variabilele sunt utilizate pentru a memora informații și pentru a face posibilă manipularea și prelucrarea datelor în cadrul programelor de calculator.

**Variabilele locale**

Variabilele locale sunt cele care sunt declarate în cadrul unui subprogram (cum ar fi o funcție sau o metodă) și sunt accesibile numai în interiorul acelui subprogram. Acestea sunt valabile numai în cadrul subprogramului în care sunt declarate. Ele nu pot fi accesate din alte subprograme sau din programul principal.

Exemplu:

function test() {

$nume = "John";

echo "Numele este: " . $nume; }

test();

**Variabilele globale**

Variabilele globale sunt cele care sunt declarate în afara tuturor subprogramelor, adică în afara funcțiilor sau metodelor. Ele sunt accesibile din întregul program. Acestea sunt valabile în întregul program. Ele pot fi accesate și modificate din orice subprogram sau din programul principal.

Exemplu:

$nume = "Maria";

function test() { global $nume;

echo "Numele este: " . $nume; }

test();

**Integrarea subprogramelor** în cadrul programelor este un aspect fundamental al programării și implică utilizarea și apelarea acestora în codul principal al programului pentru a realiza diverse operațiuni și funcționalități. Iată câteva aspecte importante despre integrarea subprogramelor în cadrul programelor:

1. Declarația și definirea subprogramelor:

Înainte de a putea utiliza un subprogram în cadrul programului principal, acesta trebuie să fie declarat și definit. Declarația subprogramului include numele său, tipul de returnare (dacă este cazul) și lista de parametri. Definirea subprogramului implică implementarea sa, adică scrierea codului care realizează funcționalitatea subprogramului.

1. 2. Apelul subprogramelor:

După ce subprogramul este declarat și definit, poate fi apelat în cadrul programului principal sau în alte subprograme. Apelul subprogramului implică utilizarea numelui său și a argumentelor corespunzătoare (dacă este cazul) într-o instrucțiune de apelare.

1. 3. Transmiterea parametrilor:

Parametrii pot fi transmiși subprogramului în diferite moduri, cum ar fi prin valoare, prin referință sau prin valoare rezultat. Aceasta permite transmiterea informațiilor către subprogram și returnarea rezultatelor către programul principal.

1. 4. Utilizarea rezultatelor subprogramelor:

Rezultatele returnate de subprograme pot fi utilizate în cadrul programului principal pentru a realiza diverse operațiuni sau pentru a obține informații necesare.

1. 5. Structurarea și organizarea codului:

Integrarea subprogramelor în cadrul programelor permite structurarea și organizarea codului în bucăți mai mici și mai ușor de gestionat. Aceasta face codul mai modular și mai ușor de înțeles și de întreținut.

**BIBLIOGRAFIE**

1. <https://danielanicolae.com/algoritmi/subprograme/>
2. <https://www.scrigroup.com/calculatoare/c/Modularizarea-programelor75192.php>
3. <https://infoas.ro/lectie/85/ce-inseamna-variabila-globala-si-locala-in-c>
4. Chatgpt