1. **Czym jest stos (stack), czym jest sterta (heap)?**

   Stos jest strukturą danych, w której dostęp do danych jest możliwy tylko do jednego elementu, będącym jego wierzchołkiem. Jest określany mianem LIFO – Last In First Out, to znaczy, że ostatni wchodzący jest pierwszym wychodzącym elementem. W STL'u posiada on następujące operacje: push (umieszczenie nowego elementu na szczycie stosu), pop (zdjęcie elementu ze szczytu stosu), empty (informacja, czy jest pusty), size (ilość elementów na stosie), top (zwraca wartość elementu przy wierzchołku). Stos jest również używany w systemach komputerowych na wszystkich poziomach funkcjonowania systemów informatycznych. Przez procesory jest stosowany do chwilowego zapamiętywania rejestrów procesora, do przechowywania zmiennych lokalnych, a także w programowaniu wysokopoziomowym.

   Sterta (heap, inaczej kopiec lub stóg) to obszar pamięci, udostępniony na wyłączność uruchomionemu programowi (procesowi), przechowuje się tam zmienne dynamiczne.

2. **Czym jest wskaźnik?**

Wskaźnik jest zmienną przechowującą adres innej zmiennej lub też zmienna wskazująca na adres innej zmiennej. Zmienną wskaźnikową w C++ tworzymy tak, jak inne zmienne zapisując jedynie operator „*". Operator ten również służy do wyłuskania wartości ze zmiennej wskaźnikowej.

3. **Czym jest wskaźnik na wskaźnik i jakie może mieć zastosowania?**

Wskaźnik wskazujący na adres innego wskaźnika jest nazywany wskaźnikiem na wskaźnik, lub podwójnym wskaźnikiem. Podwójny wskaźnik stosuje się najczęściej podczas tworzenia tablic dwuwymiarowych, by dokładnie wyłuskać wartości poszczególnych elementów takich tablic. W przypadku tworzenia zmiennych typu char, można je wykorzystać, by utworzyć listę słów a nie znaków, jak w przypadku pojedynczego wskaźnika.

4. **Na czym polega arytmetyka wskaźników?**

Na wskaźnikach można dokonywać dodawania i odejmowania, jak na zwykłych liczbach całkowitych. Warto jednak pamiętać, że zmiana adresu jest ściśle związana z typem obiektu, na który wskazuje wskaźnik. Na przykład dodanie do wskaźnika liczby 2 nie spowoduje przesunięcia w pamięci komputera o dwa bajty, tylko przesunięcie się o dwukrotność rozmiaru typu zmiennej, co w przypadku char daje przesunięcie o 2 bajty, a w float o 8. W przypadku odejmowania wskaźników wynikiem jest odległość dwóch wskazywanych wartości. Odległość zwracana jest jako liczba obiektów danego typu.

5. **Omów konwersje typów.**

Konwersja typów to konstrukcja programistyczna umożliwiająca traktowanie danej pewnego typu jak daną innego typu. Jest to bardzo przydatne podczas, gdy istnieje konieczność w programie wykonania wspólnej operacji na danych różnych typów. Konwersje mogą być rozróżniane ze względu na różne kryteria podziału. Ze względu na sposób specyfikacji wyróżniamy: jawne, ukryte (automatyczne). Natomiast według konstrukcji programistycznych: automatyczne, wymuszone, operator konwersji, podprogram, referencja.

6. **Omów rzutowanie typów.**

Rzutowanie to zmiana danych jednego typu na inny. W C++ istnieją 4 operatory rzutowania: static_cast, dynamic_cast, const_cast, reinterpret_cast. Na przykład static_cast dokonuje konwersji pomiędzy typami dającymi się w łatwy sposób rzutować (int, float, char, wchar_t, wskaźniki): „static_cast<type>(data);”.

7. **Wymień sposoby rzutowania typów w C++.**

W języku C++ możemy rzutować typy na wiele sposobów, na przykład za pomocą rzutowania niejawnego, kiedy przypisujemy, czy też wykorzystujemy w wyrażeniu wartość innego typu niż zmienna docelowa. Istnieje również rzutowanie w stylu C, to znaczy za pomocą konstrukcji: „(type) (data)”, „type (data)”, „(type) data” – type to typ danych na który rzutujemy, data to dane do rzutowania. Jeszcze jednym sposobem rzutowania jest rzutowanie za pomocą operatorów: static_cast, dynamic_cast, const_cast, reinterpret_cast.

8. **Omów różnicę pomiędzy pętlą „while” a „do… while”.**

Pętla while, podobnie jak do…while wykonuje polecenia zawarte w ciele do czasu gdy warunek w niej zawarty będzie fałszywy. Podstawową różnicą pomiędzy tymi pętlami jest to, że w while warunek sprawdzany jest przed wejściem do pętli, natomiast w do…while najpierw wykonują się instrukcje a później sprawdzony zostaje warunek. Oznacza to, że pętla while nie musi się wykonać, natomiast do…while wykona się zawsze minimum jeden raz.

9. **Omów różnicę pomiędzy statyczną i dynamiczną alokacją pamięci.**

W przypadku statycznej alokacji pamięci, pamięć zostaje przydzielona przed rozpoczęciem wykonywania programu i istnieje aż do momentu jego zakończenia. Zwalnianie pamięci następuje automatycznie po zakończeniu programu. Problemem w tym wypadku jest konieczność wcześniejszego przewidzenia wielkości obszaru pamięci jaki będzie potrzebny. Natomiast w przypadku dynamicznej alokacji pamięci dostosowuje się wielkość

zarezerwowanej przestrzeni pamięci do potrzeb programu w danej chwili. W ten sposób zaalokowaną pamięć należy samodzielnie zwolnić, by nie doszło do jej wycieków.

Eliza Zych